SAS2.0 Linux OS Driver Release Notes

Driver Version: mpt2sas-12.00.00.00-1

11/10/2011

Compatibility:

- SAS 2004, 2008, 2008_1, 2008_2, 2008_3, 2116_1, 2116_2, 2208_1, 2208_2, 2208_3, 2208_4, 2208_5, 2208_6, 2308_1, 2308_2, and 2308_3
- SSS6200

Special Notes For This Build:

- No special notes available at this time.

Current Errata

- No errata available at this time.

Driver Release Package Contents

• *Source tarball*

Source tarball is named as: mpt2sas-<version>-src.tar.gz

where:
<version> = version tag for this rpm

• *RPM Binaries:*

RPM images are named as: mptlinux-<version>-<release>-<os>.<arch>.rpm

where:
<version> = version tag for this rpm
<release> = release tag for this rpm
<os> = {rhel5, sles10}
<arch> = ia64 - Itanium Processor
x86_64 - Opteron Processor, w/ x86_64 install
i686 - i686 or later processor (Red Hat)
i586 - x86 installations (SuSE)
ppc – Power PC (64 bit)

• *Driver Update Disks*

dud images are named : mptlinux-<version>-<release>-<os>.<arch>.dd.gz

where:
<version> = version tag for this rpm
<release> = release tag for this rpm
<os> = {rhel5, sles10}
<arch> = ia64 - Itanium Processor
x86_64 - Opteron Processor, w/ x86_64 install
i686 - i686 or later processor (Red Hat)
i586 - x86 installations (SuSE)
ppc – Power PC (64 bit)

• *DKMS*

DKMS tarball is named : mptlinux-<version>-<release>.dkms.tar.gz

where:
<version> = version tag for this rpm

<release> = release tag for this rpm

This is an rpm that provide prebuilt binaries for major releases, and will compile drivers on the fly for the other kernels. Here is help on dkms: http://linux.dell.com/dkms/dkms.html

• *SRPMS*

SRPM images are named mptlinux-<version>-<release>.src.rpm

where:
<version> = version tag for this rpm
<release> = release tag for this rpm

SuSE:

i686

SLES 10

Gold (2.6.16.21-0.8)	(bigsmp, debug, default, kdump, smp, xen)
SP 1 (2.6.16.46-0.12)	(bigsmp, debug, default, kdump, smp, xen)
SP 2 (2.6.16.60-0.21)	(bigsmp, debug, default, kdump, smp, xen)
SP 3 (2.6.16.60-0.54.5)	(bigsmp, debug, default, kdump, smp, xen)
SP 4 (2.6.16.60-0.85.1)	(bigsmp, debug, default, kdump, smp, xen)

SLES11

Gold (2.6.27.19-5)	(debug, default, pae, trace, vmi, xen)
SP1 (2.6.32.12-0.7)	(default, pae, trace, xen)

x86_64

SLES10

Gold (2.6.16.21-0.8)	(debug, default, kdump, smp, xen)
SP 1 (2.6.16.46-0.12)	(debug, default, kdump, smp, xen)
SP 2 (2.6.16.60-0.21)	(debug, default, kdump, smp, xen)
SP 3 (2.6.16.60-0.54.5)	(debug, default, kdump, smp, xen)
SP 4 (2.6.16.60-0.85.1)	(debug, default, kdump, smp, xen)

SLES11

Gold (2.6.27.19-5)	(debug, default, trace, xen)
SP1 (2.6.32.12-0.7)	(default, trace, xen)

ia64

SLES10

Gold (2.6.16.21-0.8)	(debug, default)
SP 1 (2.6.16.46-0.12)	(debug, default)
SP 2 (2.6.16.60-0.21)	(debug, default)
SP 3 (2.6.16.60-0.54.5)	(debug, default)
SP 4 (2.6.16.60-0.85.1)	(debug, default)

SLES11

Gold (2.6.27.19-5)	(debug, default, trace)
SP1 (2.6.32.12-0.7)	(default, trace)

ppc

SLES10

	Gold (2.6.16.21-0.8)	(debug, default, iseries64, ppc64)
	SP 1 (2.6.16.46-0.12)	(debug, default, iseries64, ppc64)
	SP 2 (2.6.16.60-0.21)	(debug, default, iseries64, ppc64)
	SP 3 (2.6.16.60-0.54.5)	(debug, default, iseries64, ppc64)
	SP 4 (2.6.16.60-0.85.1)	(debug, default, iseries64, ppc64)
SLES11		
	Gold (2.6.27.19-5)	(debug, default, trace, ppc64)
	SP1 (2.6.32.12-0.7)	(default, trace, ppc64)

Red Hat:

i686

RHEL5		
	Gold (2.6.18-8)	(el5, el5PAE, el5xen)
	Update 1 (2.6.18-53)	(el5, el5PAE, el5xen, el5debug)
	Update 2 (2.6.18-92)	(el5, el5PAE, el5xen, el5debug)
	Update 3 (2.6.18-128)	(el5, el5PAE, el5xen, el5debug)
	Update 4 (2.6.18-164)	(el5, el5PAE, el5xen, el5debug)
	Update 5 (2.6.18-194)	(el5, el5PAE, el5xen, el5debug)
	Update 6 (2.6.18-238)	(el5, el5PAE, el5xen, el5debug)
	Update 7 (2.6.18-274)	(el5, el5PAE, el5xen, el5debug)
RHEL6		
	Gold (2.6.32-71)	(el6, el6debug)
	Update 1 (2.6.32-131.0.15)	(el6, el6debug)

x86_64

RHEL5		
	Gold (2.6.18-8)	(el5, el5xen)
	Update 1 (2.6.18-53)	(el5, el5xen, el5debug)
	Update 2 (2.6.18-92)	(el5, el5xen, el5debug)
	Update 3 (2.6.18-128)	(el5, el5xen, el5debug)
	Update 4 (2.6.18-164)	(el5, el5xen, el5debug)
	Update 5 (2.6.18-194)	(el5, el5xen, el5debug)
	Update 6 (2.6.18-238)	(el5, el5xen, el5debug)
	Update 7 (2.6.18-274)	(el5, el5xen, el5debug)
RHEL6		
	Gold (2.6.32-71)	(el6, el6debug)
	Update 1 (2.6.32-131.0.15)	(el6, el6debug)

ia64

RHEL5		
	Gold (2.6.18-8)	(el5, el5xen)
	Update 1 (2.6.18-53)	(el5, el5xen, el5debug)
	Update 2 (2.6.18-92)	(el5, el5xen, el5debug)
	Update 3 (2.6.18-128)	(el5, el5xen, el5debug)
	Update 4 (2.6.18-164)	(el5, el5xen, el5debug)
	Update 5 (2.6.18-194)	(el5, el5xen, el5debug)
	Update 6 (2.6.18-238)	(el5, el5xen, el5debug)
	Update 7 (2.6.18-274)	(el5, el5xen, el5debug)

ppc

RHEL5		
	Gold (2.6.18-8)	(el5, el5kdump)
	Update 1 (2.6.18-53)	(el5, el5kdump, el5debug)
	Update 2 (2.6.18-92)	(el5, el5kdump, el5debug)
	Update 3 (2.6.18-128)	(el5, el5kdump, el5debug)

	Update 4 (2.6.18-164)	(el5, el5kdump, el5debug)
	Update 5 (2.6.18-194)	(el5, el5kdump, el5debug)
	Update 6 (2.6.18-238)	(el5, el5kdump, el5debug)
	Update 7 (2.6.18-274)	(el5, el5kdump, el5debug)
RHEL6		
	Gold (2.6.32-71)	(el6, el6debug)
	Update 1 (2.6.32-131.0.15)	(el6, el6debug)
Oracle		
i686		
OEL5		
	Update 4 (2.6.18-164)	(el5, el5PAE, el5xen, el5debug)
	Update 4.1 (2.6.18-164.0.0.0.1)	(el5, el5PAE, el5xen, el5debug)
	Update 5 (2.6.18-194)	(el5, el5PAE, el5xen, el5debug)
	Update 5.1 (2.6.18-194.0.0.0.3)	(el5, el5PAE, el5xen, el5debug)
	Update 6 (2.6.18-238)	(el5, el5PAE, el5xen, el5debug)
	Update 6_UEK (2.6.32-100.26.2)	(el5, el5debug)
OEL6		
	RC (2.6.32-71)	(el6, el6debug)
	RC UEK (2.6.32-100.28.5)	(el6, el6debug)
x86_64		
OEL5		
	Update 4 (2.6.18-164)	(el5, el5xen, el5debug)
	Update 4.1 (2.6.18-164.0.0.0.1)	(el5, el5xen, el5debug)
	Update 5 (2.6.18-194)	(el5, el5xen, el5debug)
	Update 5.1 (2.6.18-194.0.0.0.3)	(el5, el5xen, el5debug)
	Update 6 (2.6.18-238)	(el5, el5xen, el5debug)
	Update 6_UEK (2.6.32-100.26.2)	(el5, el5debug)
OEL5		
	RC (2.6.32-71)	(el6, el6debug)
	RC UEK (2.6.32-100.28.5)	(el6, el6debug)

Major Changes For Version 11.255.02.00-1 (Phase 12 GCA)
Release Date: 11/10/2011

- CQ 228748: bump driver version

Major Changes For Version 11.255.02.00-1 (Phase 12 Beta)
Release Date: 10/12/2011

- CQ 209284: System hangs if DDOEMCLI is launched when a diag reset is in progress: Fixed the code to release the spinlock that is used to protect the raid device list before calling a function that can block. The blocking was causing a reschedule, and subsequently a different part of the driver was trying to acquire the same lock, resulting in a panic (NMI Watchdog detecting a CPU lockup).
- CQ 221329: Timed out direct-io for warpdrive results in controller reset: When an I/O request to a WarpDrive is timed out by SML and if the I/O request to the WarpDrive is sent as direct I/O then the aborted direct I/O will be retried as normal Volume I/O and which results in failure of Target Reset and results in host reset. The fix is to not retry a failed IO to volume when the original IO was sent as direct IO with an ioc status MPI2_IOCSTATUS_SCSI_TASK_TERMINATED.
- CQ 221709: System hangs when deleting raid volume that has a hot spare: (1) fix issue where Hot Spare is not reported to OS. (2) fix issue where sas_device object was partially removed but not entirely. (3) fix issue surrounding NULL pointer dereference. The underlying issues were solved and mentioned above.
- CQ 208204: CSMI: usSlotNumber is not properly getting set: Read IOC Page 1, then obtain the PCISlotNum and copy value into usSlotNumber. When PCISlotNum equals 0xFF, then only set usSlotNumber to SLOT_NUMBER_UNKNOWN. Previously usSlotNumber was set to zero because the data buffer was initialized to zero, and we were not later not setting usSlotNumber to any other value.
- CQ 208744: CSMI: bHostIndex is not getting set in the following ioctls: GET_LOCATION, GET_DEVICE_ADDRESS, and TASK_MANAGEMENT: The fix is set bHostIndex to shost->host_no.
- CQ 217028: CSMI: TUR causes CSMI pass-thru to crash: The driver was designed expecting all SSP request to be either sending or receiving data. When the memory allocation call was made with zero data length, this was resulting in oops in the memory allocation call within the kernel. To fix this issue, driver should not attempt to allocate DMA'able memory when there is no data to transfer, also the scatter gather list needs to be set to zero length sgl entry.
- CQ 209361: CSMI: HDD FW download is not working: The downloading of FW to hard disk was failing because the firmware image was not getting transferred properly from driver to hard disk. There was a bug in the memcpy command in which the data was being copied out. In addition to this fix, the driver needs to be setting the bSSPStatus to

CSMI_SAS_SSP_STATUS_COMPLETED. There was also a bug where driver was returning CSMI_SAS_SSP_SENSE_DATA_PRESENT in bConnectionStatus where it should of been bDataPresent instead. Add memory bounds checking in two IOCTLS, then when the data buffer length is too small, the driver needs to be returning CSMI_SAS_STATUS_INVALID_PARAMETER.

- CQ 224843: CSMI: fix where driver is obtaining data direction for SSP and STP passthru: The data direction is specified in two locations, namely IoctlHeader.Direction and Parameter.uFlags. This CQ requires using the IoctlHeader instead of Parameter(which we were using earlier) to know the data direction for SSP and STP Passthru.
- CQ 213453: fix memory allocation error: The amount of memory required for tracking chain buffers is rather large, and when the host credit count is big, there memory allocation failure inside __get_free_pages. The fix is to limit the number of chains to 100,000. IN addition, the number of host credits is going to be limited to 30,000 IOs, however someone can override this limitation using the command line option max_queue_depth. The algorithm for calculating the reply_post_queue_depth was changed so its equal to (reply_free_queue_depth + 16), previously it was (reply_free_queue_depth * 2).

Major Changes For Version 11.255.01.00-1 (Phase 12 Alpha) Release Date: 08/26/2011

- CQ 207483: SATA Discovery Storm fixes: The change made checks the loginfo for 0x31111000 (which is SATA Initialization Timeout code) and if so, we retry it once to check if the device is ready after the target reset initiated by the firmware, and if we receive the same 0x31111000 we return DEVICE_ERROR.
- CQ 209050: Support for greater than 2TB capacity WarpDrive: The driver is modified to allow access to the greater than 2TB WarpDrive and properly handle direct-io mapping for WarpDrive volumes greater than 2TB.
- CQ 209356: Increase max transfer support from 4MB to 16MB:This is done by changing the shost->max_sector from 8192 to 32767. The command line option max_sector upper limit was increased to 32767. End user needs to modify sysfs attribute.
- CQ 208432: Add branding string support for custom HBA's.
- CQ 217001: MPI2 Update Rev R (2.0.12)
- CQ 208770: RHEL5.7 Support: Adding KMOD, DUD, and FIXID support.

Major Changes For Version 11.00.01.00-1 (Phase 11 GCA+) Release Date: 08/12/2011

- CQ 208648: Application sending IOCTL request to driver is hanging for more than 2 minutes: There is a race in the driver where the a response is getting returned by the firmware between the time the request was sent to

firmware, and before the driver had time to go to sleep to wait for the response. When this happens, the driver is never woken up until the timeout has elapsed. This defect is occurring because the driver is initializing the completion queue in between sending the request and going to sleep. To solve the issue, the driver should initialize the completion queue prior to sending the request to firmware.

Major Changes For Version 11.00.00.00-1 (Phase 11 GCA)
Release Date: 08/05/2011

CQ 209966: bump driver version.

Major Changes For Version 10.255.04.00-1
Release Date: 07/28/2011

CQ 208718: Failure installing KMOD rpms for RHEL5.4 and RHEL5.5. The kernel API for turning on and off SRIOV is not in the kabi white list, thus the kmod rpm fails to install. The fix is to not compile in SRIOV support for RHEL5.4 and RHEL5.5.

Major Changes For Version 10.255.03.00-1
Release Date: 07/22/2011

- CQ 206770: Driver is not reporting devices that don't support REPORT_LUNS: When a device that doesn't support REPORT_LUNS is used, the scsi mid layer will invoke sequential lun scan. When there is sequential lun scan, eventually the inquiry is going to fail, and scsi mid layer will remove that instance of the lun. A bug in the driver is resulting in the entire target getting deleted when the first failing LUN is detected. The driver is setting the sas_device->starget to NULL from slave_destroy callback for LUN=1 even though LUN=0 exist. This is resulting in entire target getting deleted. To resolve the issue, the driver should only set sas_device->starget to NULL when all the LUNS have been deleted from the slave_destroy. On SLES11 with `scsi_mod.scan=async` on the kernel command line, the driver hangs for the same TAPE device. Its not possible to delete device when asyn scanning is enabled. The fix is prevent devices getting deleted at driver load time when asyn scanning is activated.

Major Changes For Version 10.255.02.00-1 (Phase 11 beta)
Release Date: 07/05/2011

- CQ 204262: CSMI SSP and STP Passthru's are not working: The routine that is checking whether the port and phy identifier is valid is failing for SSP/STP passthrus because its expecting the phy identifier to be 0xFF meaning invalid. The routine instead needs to be expecting the phy number to be valid for up to the max phys. The resolution is to create two seperate routines for checking the port and phy indentifier, one is for SMP and the other for SSP/STP. The routine for SSP/STP will check the phy

indentifier making sure its in expected range. Whereas SMP passthru the phy indentifier should be 0xFF.
- CQ 204597: Modify the Makefile for DKMS kit so CPQ_CIM is defined.
- CQ 202822: DUD install is not happening. The error message that comes out while doing a DUD install using a USB - " Cannot find /tmp/drivers/rpms/i386/repodata, bad driver disk". The resolution is to use the kernel_version macro defined explicitly in spec file to generate individual kmods.
- CQ 204170: Repackage SAS2 Phase 10 Driver with LDPK 1.2-04. Received the updated FIXID LKPD from the customer. Extracted the kit using rpm2cpio/cpio. Compared the differences between the new and existing kit, and the only change was the ibm-driver-tools RPM for RHEL. The RPM was upgraded from version 132 to 134. The change in our build kit is to replace the 132 version of the RPM with the 134.

Major Changes For Version 10.255.01.00-1 (Phase 11 alpha)
Release Date: 05/25/2011

- CQ 196681: The release deliverable sub-folders are renamed to contain the Operating System.
- CQ 199007: Adding Phase 11 MPI Headers into the Linux Driver
- CQ 194591: RHEL6.1 support
- CQ 194590: SLES10 SP4 support
- CQ 183403: adding branding support to identify Ramsdale adapter
- CQ 177148: Request to change Linux driver to replace IOs terminated with DID_RESET with DID_SOFT_ERROR to avoid infinite resets
- CQ 189345: Add Enhancement to better handle DEAD IOC (PCI-E LInk down) error condition: Change has done in two steps. Detection of the dead IOC, removal of dead IOC
- CQ 193588: Adding SSS6200 support.

Major Changes For Version 10.00.00.00-1 (Phase 10 GCA)
Release Date: 05/11/2011

- CQ 194596: bump driver version

Major Changes For Version 09.255.08.00-1
Release Date: 05/06/2011

- CQ 193377: Use a different method for blocking IO request from SCSI Mid Layer when using the SysFS shost attribute called task_management: There is a SysFS attribute called task_management. It allows a backdoor interface for sending task management request to the driver like target reset, lun reset, abort task, and abort task set. When these methods are used, we need to freeze the IO queues coming from the SCSI Mid Layer above. The existing method for doing this is setting the ioc->shost_recovery flag, which will return SCSI_MLQUEUE_HOST_BUSY to upper layers while that flag is set. This will allow the IO to be requeued

later. However with this test case, the target reset is generating a Broadcast AEN event. From the Broadcast AEN event in the driver, we are using the flag ioc->shost_recovery to determine whether to unblock. When ioc->shost_recovery is set, the hard reset handler will eventually unblock, we don't want to be unblocking the devices when there is an actual hard reset, or when ioc->shost_recovery is set, as is the case at the end of the Broadcast AEN handler. From the SysFS attribute task_management, we need to use a different method for blocking IO coming from SCSI Mid Layer. To address this issue, we remove setting ioc->shost_recovery flag, and replace it with using scsi_block_requests(ioc->shost) and scsi_unblock_requests(ioc->shost).

Major Changes For Version 09.255.07.00-1
Release Date: 05/04/2011

- CQ 193687: Don't allow IOCTLS to be processed when driver is loading, and fix the two underlying panics: While port enable is active, the driver is receiving a diagnostic reset from application. This is causing the port enable to timeout which results in driver unload itself. When the driver unloads itself, there is a panic because driver is taking on commands after it has unloaded itself. Once that panic is solved, there is another panic which occurs when loading the driver. The 2nd panic is due to driver not releasing reference count when unloading due to driver load failure. (1) Modify IOCTLS so they are returning -EAGAIN while the driver is in the middle of loading. This will result in application eventually retrying the request once the driver is fully loaded.
  (2) Modify driver so its setting the remove_host flag inside mpt2sas_base_hard_reset_handler when port enable failed during driver load time. This will result in all request being returned with DID_NO_CONNECT. This willl prevent a panic when IO is sent to driver between diagnostic reset and driver unloading itself.
  (3) Modify driver so its calling scsi_host_put(shost) when its failed to load. This will decrement reference count, therefore if when the driver is loaded again, a panic will be averted.

Major Changes For Version 09.255.06.00-1
Release Date: 05/02/2011

- CQ 190530: Add support in asynchronous scanning logic to handle fault during discovery when loading the driver. If there is a fault occurring during discovery(port enable) then the driver will attempt to do a host reset. During that host reset, there will be another discovery(port enable) sent, and the same fault will occur again. So this will lead to the driver processing the same fault over and over again. Ultimately if this problem occurs, the customer should look at upgrading Controller Firmware. The fix in the driver is to prevent never ending processing faults when this

situation occurs at driver load time. The new behavior is to return DID_NO_CONNECT for all devices once the fault occurs. The change set includes (1) Separate the port enable handling from the base_cmds so port enable is sent not effecting any other commands sent from the base modules, this includes creating a completion routine unique only for by port enable, and creating port_enable_cmds and port_enable_cb_idx in the ioc. (2) When host reset occurs, set a flag so the asynchronous scanning logic stops polling the driver, for older kernels, wake up the sleeping port enable thread that originally sent the request. (3) From reset handling, set the remove_host flag which cause DID_NO_CONNECT to be returned for all devices from the queuecommand entry point. (4) Added port_enable_failed flag in ioc to indicate port enable failed. This flag is checked from host reset so another port enable is averted if the previous one failed.

- CQ 183910/CSET 192562 : Allocate only one msix vector for the controllers that don't support multi-reply queues and NUMA IO. This is to resolve issues seen where interrupts are not routed when too many msix vectors are reported to the linux kernel for the non-numa cards.
- CQ 189694/CSET 192566 : Add per lun delete flag: Address test case where cable is removed and added back within the DMD timer expired timeout. For this particular case, it involves at least one one lun within a target failing the TURs sent following links up event. The observed failure is LUN zero is returning sense 0x05/0x25/0x00 meaning "LOGICAL UNIT NOT SUPPORTED", this returns DEVICE_ERROR in the driver, and the per target delete flag is set to one, then driver is returning DID_NO_CONNECT for all luns within the target. To solve this, we need a per lun flag to indicate the LUN is no longer responding commands.
- CQ 192418: Fix bug surrounding reading raid configuration information for inactive volumes. The driver is calling a routine to obtain the device handle for an inactive volume. This routine is failing for inactive volumes because the Action was set to MPI2_CONFIG_ACTION_PAGE_READ_CURRENT when asking for RAID_CONFIG page zero. This will only return active volumes, not including inactive. Due to this failing, the driver doesn't configure the physical drives. When the physical drives are not configured, the OS would tear down the scsi_device object, however the driver was keeping around an instance of sas_device object internally. Due to the sas_device object dangling around, the driver would eventually panic when the inactive volume was deleted. Fix bug in mpt2sas_config_get_volume_handle. The driver was setting the action to MPI2_CONFIG_ACTION_PAGE_READ_CURRENT, which only returns active volumes. In order to get info on inactive volumes, the driver needs to change the action to MPI2_RAID_PGAD_FORM_GET_NEXT_CONFIGNUM, and traverse each config till the iocstatus is MPI2_IOCSTATUS_CONFIG_INVALID_PAGE returned. Also added a

change in the driver to remove the instance of sas_device object when the driver returns "1" from the slave_configure callback.

- CQ 192903: PCIe EEH fix for SLES10: Add support for SLES10 in mpt2sas_base_map_resources so the pci state is backed up from mpt2sas_base_map_resources. Currently this only occurs for SLES11 an RHEL6, but implemented further down the routine after resources are mapped. For SLES10 this needs to occur prior to mapping in resources.
- CQ 192786: The device driver doesn't compile under RHEL6.1: This issue is due the change_queue_depth API change from 2.6.33 kernel. In older kernels change_queue_depth API is using in two input parameters, whereas the newer kernels are using three parameters. The problem is Red Hat back ported this support to 2.6.32 kernel, whereas the UEK version of 2.6.32 is missing this support. Therefore the driver is going to fail to compile in both version of the 2.6.32 due to the API differences. In the RHEL version of the Makefile, they have defined RHEL_RELEASE_CODE. So we can use this define to distinguish between the OEL UEK version of 2.6.32 versus the Red Hat version of the same kernel version.

Major Changes For Version 09.255.05.00-1
Release Date: 04/18/2011

- CQ 188119: Stack trace seen in message log after cable pull + host reset. Resolve a deadlock between hot plug worker threads and host reset context. This is due to driver reporting a device missing to the OS then the OS sending a SYNC_CACHE request to driver while the IO queues are locked due to host reset. A recent change in the driver pushed the port enable wakeup to the hot plug worker threads, so the host reset context will not complete till the worker threads processed the fake port enable wakeup event. To fix the issue, the driver will be waking up the port enable context immediately when the driver receives the reply message, instead of waiting on the hot plug worker threads. Also, there was driver change in _scsih_remove_device to unblock the devices had there not been hotplug event to unblock them. This code was not being called because _scsih_prep_device_scan was setting the deleted flag to "1" for all devices.
- CQ 187701: The configuration pages are failing if raid volume is configured while issuing a host reset. Add error checking in slave_configure to check for configuration pages failing, and return "1" so the device is not configured. The config pages are failing if raid volume is configured while issuing a host reset, thus driver is reading stale data and proceeding to attempt to add. The fix is to return error so the volume is not configured. Its okay if the driver returns with error from slave_configure because from the post reset handling, the driver is scanning for new volumes, and the volume is eventually added.

Major Changes For Version 09.255.04.00-1
Release Date: 04/11/2011

- CQ 188119: Devices are not removed after cable pull: In a recent fix, the driver behavior changed where the device handle was set to MPT2SAS_INVALID_DEVICE_HANDLE prior to unblocking the device. Due to this change, the driver was not unblocking the device since the device handle was not defined, hence the SCSI mid layer went into deadlock when SYNC_CACHE was sent.
- CQ 188923: If devices are behind two deep cascaded expanders, they don't get added if the "cable add" is done while host reset is active. Add support to update expander phys link information across host reset. Since this was not being updated, the child device could not be added because it was unable to locate its instance within the parent expander phys.
- CQ 188949: Ensure that the initial reference tag is passed on to the HBA firmware for DIF Type 2 devices. Patch provided by Martin Petersen.
- CQ 188953: Reduce cache misses in command submission path: In a high-IOPS workload, mpt2sas_base_get_smid_scsiio shows up in the top 20 cache misses. This is because the data structure used for allocating SMIDs is cache unfriendly; freed SMIDs are placed on the tail of the list, guaranteeing a cache miss by the time we allocate it again. By placing the freed SMID at the head of the list, we increase the likelihood of it being cache-hot when it's used again. This patch provided by Matthew Willcox.
- CQ 188951: prevent heap overflows and unchecked reads: At two points in handling device IOCTL's via /dev/mpt2ctl, user supplied length values are used to copy data from user space into heap buffers without bounds checking, allowing controllable heap corruption and subsequently privilege escalation. Additionally, user supplied values are used to determine the size of a copy_to_user() as well as the offset into the buffer to be read, with no bounds checking, allowing users to read arbitrary kernel memory. This patch provided by Dan Rosenberg.

Major Changes For Version 09.255.03.00-1
Release Date: 03/31/2011

- CQ 186401: Failure message displayed in the log messages during host reset while removing devices: Following host reset, the driver is doing a device scan to determine whether any devices were added while host reset was active. This is accomplished by walking the configuration pages. It seems that firmware is reporting a valid sas device page for given device which was recently pulled. The parent device handle reported in that configuration page is invalid, that is 0x5441. When the driver sends a configuration page request for the parent handle of 0x5441, the firmware returns 0x22 ioc status, hence the error message seen in the log. The fix is to inhibit the warning message in _scsih_get_sas_address when the MPI2_IOCSTATUS_CONFIG_INVALID_PAGE ioc status is returned.

- CQ 185626: Drives are not getting removed when the enclosure cable is pulled out from the expander when Port Reset is issued: Allow the FW/Driver device removal handshake to occur when ioc->shost_recovery is set. The current design is to not allow the handshake to occur while the host reset handling is active, which is when ioc->shost_recovery is set. However per this test case, device removal events are occurring while port enable is active during host reset context. The change is to check the ioc_state instead of ioc->shost_recovery to determine whether to send the handshake during host reset context.
- CQ 186681: Possible deadlock bug in Phase9 SAS2 Linux driver: The customer reported deadlock occurring when multiple devices are being removed simultaneously. This is occurs when between one target removed from the driver hotplug kernel worker threads, and the other target removed in the FW/Driver device removal handshake. The deadlock is between two spin locks, between the shost->host_lock and driver ioc->sas_device_lock. The fix is to rearrange the code in the FW/Driver device removal handshake so the ioc->sas_device_lock is not occurring when the shost->host_lock is taken.
- CQ 186643: Removal and insertion of driver module on RHEL 6.0: Prevent driver from unloading when its still loading. This issue only affecting Red Hat, not SuSE.
- CQ 185979: Chip Resets results in repeat timeouts: This test case involves controllers with NUMA support. Following diag reset, the driver is not reinitializing the ReplyPostHostIndex for every reply queue, thus FW/Driver get out of sync where the replies are located. Also removed the HW workaround for backing up and restoring the MSIX vectors in the PCIe MSIX vector capabilities registers, the workaround was only for Falcon A0 part.
- CQ 186567: Running the load_diag_trace_on.sh and diag_rest with 30 secs delay make the driver to crash after some time: The issue surrounds the load_diag_trace_on.sh driver sending a request to the driver while a diag reset is active. If this occurs before ioc is initialized, the message queues will not be setup, thus resulting in a 1550 fault. To fix the problem: (1) We have moved the "ioc_reset_count" counter so its incremented after the host reset has completed bringing up the controller. This will prevent the script from sending the request too early. (2) We have added a sanity check in the Sysfs so the script will not be allowed to send a request to driver while driver is busy with recovery. (3) All the diag buffer code will be changed to use an unique message frame callback index. The previous implementation was having diag buffer code share the same callback index with IOCTLs. So if there was a timeout with IOCTLS, it would not be possible to send a RELEASE prior to diag reset because IOCTLs was owning the callback index due to the timeout. With this change, this will insure the RELEASE will be sent.

Major Changes For Version 09.255.02.00-1

Release Date: 03/13/2011

- CQ 184402: Power PC - first request sent through message queues is timing out: We have to revert to sending two separate 32bit pci writes, accompanied with spin locks.
- CQ 184414: Hang when there is smart error on IBM platform.: Driver was a sending a SEP request during interrupt context which required to go to sleep. THe fix is to rearrange the code so a fake event MPT2SAS_TURN_ON_FAULT_LED is fired from interrupt context, then later during the kernel worker threads processing, the SEP request is issued to firmware.
- CQ 184696: Call Trace seen during hotplug & reconnect during Boot time: A device is in the process of being detected at driver load time then all of sudden is removed. The oops is due to the driver freeing memory for the new device, then placing sas_device object on a link link called ioc->sas_device_list. Then the next device that is added to the list is accessing freed memory, thus causing the oops. To fix this issue, the device that is no longer present should not be added to the list. So the code in _scsih_probe_sas() is rearranged as such so the devices that failed to be detected are not added to the list.
- CQ 184960: Ph10:DUD installation is failing on OEL6 32 bit: Modified Build.rules so it creates a symbolic link on the DUD "i386 -> i686".
- CQ 185002: Adding support for RHEL6 Power PC kmods/dud

Major Changes For Version 09.255.01.00-1
Release Date: 03/03/2011

- CQ 177319: CSMI IOCTLS four part ID check: Adding new routine for checking the four part pcids for five customer branded controllers, and only allowing these controllers to work with CSMI ioctls. For any other controller, the driver will return -EPERM when sending CSMI, meaning permission denied.
- CQ 17683: Add support to turn on SRIOV support: (1) Detect if controller firmware is supporting SRIOV by checking for the presence of the PCI Capability Registers for SRIOV at offset 16. This is defined in kernel as PCI_EXT_CAP_ID_SRIOV. (2) Get max supported VFs from PCI Configuration Registers at offset 0x15E (3) Implement command line option called max_vfs. This allows end user to set the maximum virtual functions. (4) Use the max between #2 and #3 above (5) Call the kernel API pci_enable_sriov(). When calling this API, need to pass the number requested Vfs
- CQ 180358: Upgrading to Phase 10 MPI headers. Header Change log: (1) Added Temperature Threshold Event, IO Unit Page 8, and IO Unit Page 9 to support temperature sensor monitoring and reporting. (2) Added SendHostMessage message and Host Message Event. (3) Added SAS Notify Primitive Event. (4) Added ProxyVF_ID field to Configuration

Request message. In various configuration page sections, updated how each page type behaves in regards to virtual functions. (5) Added IO Unit Page 10 for specifying the credit allocation.

- CQ 176830: Fast Load Support: (1) Asynchronous SCSI scanning: This will allow the drivers to scan for devices in parallel while other device drivers are loading at the same time. This will improve the amount of time it takes for the OS to load. (2) Reporting Devices while port enable is active: This feature will allow devices to be reported to OS immediately while port enable is active. The old implementation is waiting for port enable to complete, and then report devices. This feature is only enabled IT firmware configurations when there are no boot device configured in BIOS Configuration Utility, else the driver will wait till port enable completed to report devices. For IR firmware, this feature is turned off. This enhancement is to address large SAS topologies (>100 drives) when the boot OS is using onboard SATA device, in other words, the boot devices is not connected to our controller. (3) Scanning for devices after diagnostic reset completes: A new routine is added by the enhancement; this will scan the expander pages, IR pages, and sas device pages, then reporting new devices to SCSI Mid layer. It seems the driver is not supporting adding devices while diagnostic reset is active. Apparently this is due to the sanity checks on ioc->shost_recovery flag throughout the context of kernel work thread FIFO, and the mpt2sas_fw_work.
- CQ 181786: Fix compile error in OEL 2.6.32 kernels: Changed the KERNEL_VERSION check so the "device queue ramp up code" is going to compile for (>= 2.6.33) or the SLES11 SP1 kernels. For RHEL6 you see the following harmless warning when compiling the driver.
- CQ 181789: Adding support for displaying customer branding info.
- CQ 182313: Adding SRIOV command line option: Adding command line option to turning on SRIOV. By default SRIOV will be off when command line option is not used. Added a flag in struct MPT2SAS_ADAPTER for tracking whether SRIOV has been turned on. Added sanity check for the command line option "max_vfs" to insure that when its set to zero, the minimum max_vfs is set to 1.
- CQ 176888: Oracle OEL 5.6 and 6.0, and the UEK (unbreakable kernel) Support: (1) Added new OEL6 build kit based on the RHEL6 kit. (2) Added DUD/KMOD support for the native RHEL5.6 and RHEL6 kernels inside the OEL kits (3) For UEK kernels, only the KMODs rpms are provided, not the DUDs. The reasoning behind this is the OEL installation are using the native Red Hat kernels, not UEK. (4) There was a workaround in both kits in generating the UEK rpms. For OEL5.6, we need to bypass the ksym checking in /usr/lib/rpm/redhat/find-requires, and in OEL6.0, we need to modify /usr/lib/rpm/redhat/macros so the %kernel_source is set properly, so it needs to query using kernel-uek-headers instead of kernel-headers.

Major Changes For Version 09.255.00.01-1

Release Date: **02/10/2011**

- CQ 159970: Add support for v1.02 of the CSMI Specification.
- CQ 177319: HP CSMI IOCTLS four part ID check - Adding new routine for checking the four part pcids for five HP branded controllers, and only allowing these controllers to work with CSMI ioctls. For any other controller, the driver will return -EPERM when sending CSMI, meaning permission denied.

Major Changes For Version 08.255.07.00-1
Release Date: **02/10/2011**

- CQ 177161: Adding support for displaying customer branding information for the Mustang controller.

Major Changes For Version 08.255.06.00-1
Release Date: **02/04/2011**

- CQ 174579: Spitfire issuing 0x620f fault while IOs and TM are running: The implementation of writeq in 32bit kernels for RHEL6 is incorrect: There is no spin lock around two separate 32 bit pci memory writes, thus introducing a race condition where two separate cpus are writing to the request descriptor at the same time. To fix this issue, we need to always acquire a spin lock when doing the 64 bit pci memory write.
- CQ 175941: Hibernation is not happening properly in RHEL 6 and SLES 11 operating systems: The scsi midlayer is deadlocked when devices are removed from the driver pci_driver->shutdown handler. To fix this issue, we need to revert the shutdown handler back to just tearing down kernel threads, and sending the ir_shutdown request. Also, from scsih_qcmd, we add check for the ioc->remove_host flag, and return DID_NO_CONNECT.
- CQ 176480: fix debug messages in _scsih_determine_disposition: The scsi_state and scsi_status are swapped.

Major Changes For Version 08.255.05.00-2
Release Date: **01/25/2011**

- CQ 174974: RHEL5.6 - adding support for the 2.6.18-238 kernel in build kits.
-

Major Changes For Version 08.255.05.00-1
Release Date: **01/17/2011**

- CQ 171675: The name for installed kmp rpms of drivers are too big and ambiguous: Followed the same model as GEN1 build kits with regards to the naming of the rpm. Here is the change set: (1) Modified the make_kmp script so it can sed the subpackage-spec to contain the exact kernel which is getting built. (2) Modified the subpackage-spec so it

contains a define called "kernelname". This will be defined to the exact kernel which is getting built by the make_kmp script.
- CQ 173329: IBM Fixid update: Upgraded the ibm-driver-tools to version 132 for RHEL and version 130 for SuSE.
- CQ 168468: problems with the KMP spec file: Changed BuildRoot from %{_tmppath}/{%name}-%{version}-build to %{_tmppath}/%{name}-%{version}-build
- CQ 172460: Target Mode Driver is having R/W errors: (1) pass the msix_index via parameters to the target mode driver ISR routine for handling new command buffers. (2) Add hook into watch dog timer routine so target mode driver can poll outstanding commands for status.

Major Changes For Version 08.255.04.00-1
Release Date: 12/30/2010

- CQ 171694: Driver panic for controllers not supporting NUMA: The driver is panicking from base_interrupt when there is task management request sent for controllers not supporting NUMA. The mpt2sas_base_flush_reply_queue routine is called for task management request to flush the IO for all the other queues. When NUMA is turned off, then multi-reply queues are not setup, however the routine mpt2sas_base_flush_reply_queue is still calling reply_q objects. Since these reply queue objects were not initialized, it would panic in the ISR due to NULL pointer dereference. To fix this issue, the driver needs to check whether NUMA is turned on prior to traversing the reply queue link list.

Major Changes For Version 08.255.03.00-1
Release Date: 12/28/2010

- CQ 166704: Properly handling of target reset in multi-initiator environment: (1) Clean up in broadcast change handling: Need to look at the status of each task management request, and retry the TM when there are failures. Need to quiesce IO so the driver doesn't take on more IO request while its in the middle of sending TM request to firmware . Add support to keep track of how many pending broadcast AEN events are received while the broadcast handling is active, then loop back at the end of this routine if there were any events received. (2) Clean up in mpt2sas_scsih_issue_tm routine: Make sure proper status is returned when host reset fails. Clean up sanity checks near end of routine, insuring all outstanding IO were completed.
- CQ 171271: From the broadcast AEN processing, the driver is processing the task management reply immediately following both Query Task Abort Task completions. So if OS generated TM is allowed to be processed while broadcast AEN is active, then it could clear the memory associated to the mpi_reply and the Broadcast AEN processing would be taking the incorrect action. If OS generated TMs were allowed with Broadcast AEN

is active, then its possible race where the TM be sent in between the Query Task and Abort Task from broadcast AEN processing. We want both Query Task and Abort Task to complete for given IO. The fix is to hold a TM mutex during entire broadcast AEN processing. The same mutex is held for OS generated TMs.

- CQ 171274: Task aborts are failing when NUMA is enabled: Due to multi-reply queues, it is possible for task management reply and the aborted SCSI_IO replies complete on separate queues. When the replys are occurring simultaneously on separate interrupts, its a possibility that the driver could receive the TM reply before all the pended SCSI_IO complete. When that occurs, the sanity checks at the end of TM processing would fail, and the driver returns FAILED status up the stack to the scsi mid layer, and error recovery is escalated to the next level. If the escalation continues to host reset, then all the devices get off-lined. The driver fix: (1) Issue all task management request on reply queue zero. (2) From the task management interrupt handler,the driver flushes out all the other reply queues. This is done by traversing each reply queue and calling its main interrupt handler. This will insure the SCSI_IO reply are processed before the TM reply is waking up its sleeping thread from processor context. (3) From the main interrupt handler add an atomic flag which will indicate whether the ISR is already active for the respective reply queue. If the ISR is active, then return doing nothing. This check is required so the flushing of the reply queues don't occur when the interrupt handler is active processing replys from another processor.

Major Changes For Version 08.255.02.00-1
Release Date: 11/19/2010

- CQ 166544: PH9 Linux Drivers: adding support to retrieve firmware diagnostic trace buffers between every diagnostic reset (and FAULTs): The current problem we face is the ring buffer or trace buffer is only good since the last host reset. This enhancement handle test cases involving multiple diag resets, where firmware engineers are asking for trace buffer between each host reset. Additional changes in the driver are as follows: (1) inhibit incrementing the ioc->ioc_reset_count counter when the driver is unloading (2) change the ioc->diag_buffer_status to MPT2_DIAG_BUFFER_IS_RELEASED for the case when there is a FAULT (3) change the ioc->ioc_reset_count SysFS attribute from "%08d" to "%d", as the script was failing after 8 iterations of resets. (4) change the pulling of the trace buffer size from 4096 to 4095 (because hit was hitting an warning in the kernel)

Major Changes For Version 08.255.01.00-1
Release Date: 11/17/2010

- CQ 153119: Set Max Sector Count of SAS2 MPT Linux Driver: Request is to have the capability to override the default max_sectors setting at load time, taking max_sectors as an command line option when loading the driver. The setting is currently hard-coded in the driver to 8192 sectors (4MB transfers). If max_sectors is specified at load time, minimum specified setting will be 64, and the maximum is 8192. The driver will modify the setting to be on even boundary. If max_sectors is not specified, the driver will default to 8192.
- CQ 162850: System panics when rebooting in a multi-pathing configuration. The root cause is due to the mult-pathing driver is sending additional request to the driver after the PCI shutdown notifier is called. From the shutdown routine, the driver is freeing all the memory allocated associated to sending IO request, so if there additional IO request arriving after this point, the host will panic. To solve this issue, the driver needs to report all the devices missing to the OS prior to freeing its memory.
- CQ 165965: Phase 9 revision Q header update
- CQ 156055 – Big Endian issue on 32bit PPC: (1) Removed the BITS_PER_LONG in _base_send_ioc_init as this code was failing to compile on 32bit PPC. This code was replaced by using the cpu_to_le64 API and u64 typecasting for the 32 bit fields. (2) Fixed all the BIG ENDIAN issues by installing sparse, and compiling with the C=2 CF="-D__CHECK_ENDIAN__" option.

Major Changes For Version 08.00.00.00-3
Release Date: 11/10/2010

- CQ 164577 – DUD installation fails on SLES10 SP3: The SLES10 duds had the incorrect path to the updated KO files needed for installation.

Major Changes For Version 08.00.00.00-2
Release Date: 11/08/2010

- CQ 163957 – ph8: sles10 duds broken: During Phase 8, the driver update disks were changed to ISO format so they would have enough room to contain RPMs. It seems the DUDs are missing the RPMs. The problem is due to build script ordering of the running of the kits. Since the DUD depends on the KMP kit, the DUD kit needs to be executed after KMP, thus resulted in the RPMS missing. The solve this, the DUDs kit needs to be run last in the list of kits.

Major Changes For Version 08.00.00.00-1
Release Date: 10/28/2010

- CQ 162842 – Phase 8 GCA bump driver version.

Major Changes For Version 07.255.08.00-1
Release Date: 10/07/2010

- CQ 157528 – OEL5.5 installation failed with DUD. The OEL build kit was separated from the RHEL5 kit, and the actual build done on a OEL based system. Previous builds were being done on RHEL5 for OEL, and this was creating signing incompatibilities with the prepackaged binaries.
- CQ 155847 - IR shutdown send/complete messages are not occurring : In a recent bug fix, code was added to remove the volumes from driver callback pci->remove routine. When this change was made, the device handles were no longer present in the driver by the time the _scsih_ir_shutdown was called. So from the scsih_ir_shutdown routine, the driver was not able to send the request to FW since it couldn't locate the device handles. To fix this issue, the driver is going to need to call the _scsih_ir_shutdown prior to reporting the volumes missing from the OS, hence the device handles are still present.

Major Changes For Version 07.255.07.00-2
Release Date: 09/21/2010

- CQ 147801 – RHEL6 RC1 (2.6.32-71) build support (KMOD/DUDS/FIXID)

Major Changes For Version 07.255.07.00-1
Release Date: 09/09/2010

- CQ 149409 : SLES11 SP1 : dkms rpm doesn't work when the bundled driver is a newer driver version: The bundled driver in SLES11 SP1 is based on Phase 4. If the customer is using a version of the driver older than the bundled version, the dkms will not install the driver. This change is to pass the –force command line option for "dkms install" in order for the installed driver from the package to take
- CQ 147987 Linux Driver-KMP install using DKMS needs change in code: The "mkkmp" command was failing due to the kmp spec file having incorrect info for the BuildRequires line. This was solved by changing the line to "BuildRequires: kernel-source kernel-syms"
- CQ 149127 - Add support for Customer specific Identification: Report branding messages when device driver loads, based on specific customer subsystem vendor and device Ids.
- CQ 144099 - False timeout after hard resets: There were two issues which leads to timeout. (1) Panic because of invalid memory access in the broadcast asyn event processing routine due to a race between accessing the scsi command pointer from broadcast asyn event processing thread and completing the same scsi command from the interrupt context. (2) Broadcast asyn event notifcations are not handled due to events ignored while the broadcast asyn event is activiy being processed from the event process kernel thread. In addition, changed the ABRT_TASK_SET to ABORT_TASK in the broadcast async event processing routine. This is less disruptive to other request that generate Broadcast Asyn Primitives besides target reset, those for for example are clear reservations, microcode download, and mode select.

- Q 152497 - Kernel Panic during Large Topology discovery: There was a configuration page timing out during the initial port enable at driver load time. The port enable would fail, and this would result in the driver unloading itself, meanwhile the driver was accessing freed memory in another context resulting in the panic. The fix is to prevent access to freed memory once the driver had issued the diag reset which woke up the sleeping port enable process. The routine _base_reset_handler was reorganized so the last sleeping process woken up was the port_enable.
- CQ 154993 - basic code cleanup's: (1) _base_get_cb_idx and mpt2sas_base_free_smid were reorganized in similar fashion so the order of obtaining the cbx and smid are scsiio, hi_priority, and internal. (2) The hi_priority and internal request queue struct was made smaller by removing the scmd and chain_tracker, thus saving memory allocation. (3) For scsiio request, a new structure was created having the same elements from the former request tracker struct.
- CQ 155015 - fix RHEL6 compile error: MAX_PHYS_SEGMENTS is no longer defined in newer kernels. It will be replaced with SCSI_MAX_SG_SEGMENTS.

Major Changes For Version 07.255.06.00-1
Release Date: 08/24/2010

- CQ 97380 - NUMA IO support: (1) Create the new structure adapter_reply_queue to contain the reply queue info for every msix vector. This object will contain a reply_post_host_index, reply_post_free for each instance,msix_index, among other parameters. We will track all the reply queues on a link list called ioc->reply_queue_list. Each reply queue is aligned with each IRQ, and is passed to the interrupt via the bus_id parameter. (2) The driver will figure out the msix_vector_count from the PCIe MSIX capabilities registers instead of the IOC Facts->MaxMSIxVectors. This is because the firmware is not filling in this field until the driver has already registered MSIX support. (3) If the ioc_facts reports that the controller is MSIX compatible in the capabilities, then the driver will request for multiple irqs. This count is calculated based on the minimum between the online cpus available and the ioc->msix_vector_count. This count is reported to firmware in the ioc_init request. (4) New routines were added _base_free_irq and _base_request_irq, so registering and freeing msix vectors were done thru simple function API. (5) THe new routine _base_assign_reply_queues was added to align the msix index's across cpus. This will initialize the array called ioc->cpu_msix_table. This array is looked up on every MPI request so the MSIxIndex is set appropriately. (6) A new shost sysfs attribute was added to report the reply_queue_count. this is needed for new script set_affinity. (7) A new script set_affinity was added in the source code. This script will set the affinity cpu mask, so the interrupts occur on the same cpu that sent the original request.

- CQ 147817 - Modified the set_affinity script to handle the case where there are more core processors than available msix vectors: The driver will align the MSIX vector assignment so adjacent processor cores are sharing the same interrupt. The modification was done both the driver source and script.
- CQ 152211 - Driver oops when loading under sles10: The root cause is due to CQ 147144, and the corresponding code change in mpt2sas_base_map_resources where its calling pci_save_state to backup the pci configuration. Apparently in the older kernels this function call is corrupting memory. The code change is to only enable this for SLES11 and RHEL6. The same change was made for pci_restore_state function call in the PCIe error recovery callbacks.

Major Changes For Version 07.255.05.00-1
Release Date: 08/19/2010

- CQ 147803 - Phase 8: Add OEL support: (1) Support was added for creating OEL driver update disks and KMOD rpms. The modification was in the RHEL5 ddiskit. It involved adding new kversion files for the following OEL kernels: (2.6.18-164.el5, 2.6.18-164.0.0.0.1.el5, 2.6.18-194.el5, and 2.6.18-194.0.0.0.3.el5). (2) The kernel modules and devel rpms were installed RHEL5 build systems for the following (2.6.18-164.0.0.0.1 and 2.6.18-194.0.0.0.3). Please note that the 2.6.18-164.el5 and 2.6.18-194.el5 are identical to RHEL 5.4 and 5.5 so they didn't need to be installed. (3) Added the make_release script in the SLES11 kit for generating the final deliverable file called mpt2sas-release.tar.gz.
- CQ 147802 - Phase 8: fixid changes: (1) removed fixid from the RHEL5/SLES10 kits (2) sles11 kit modified so deliverables are pulled from the other sections of the kit. With this change, builds are going to complete in half the time because rpms and duds are not generated twice. This also makes it easier to support fixid. (3) ldkp v1.0-20 was incorporated into the kit (4) ABI checking is going to be done from a separate enhancement later
- CQ 151006 - Revision P header update: (a) Added enable/disable SATA NCQ operations to SAS IO Unit Control Request. (b) Modified Host Based Discovery Action Request message format. (c) Removed Device Path bit from IO Unit Page 1 Flags field. (d) Added description of ChainOffset field for Diagnostic Data Upload Tool. Chaining is not allowed.
- CQ 147799 - convert Red Hat Driver Update Disks into ISO format, and add KMOD RPMs: (1) upgraded to v1.0.0 ddiskit, (2) the make_dud script was modified so it does `make diskiso`, which generates duds in ISO format (3) The kmod spec file and kmodtools are bundled in the kit, which is required for creating the rpms. (4) the make_dud script is modified so rpms are copied to release/kmods- folder, therefore the kmodkit can be removed (we don't need to create kmods twice) (5) the release/srpms- folder in removed. Instead the source rpm will be found in release/rpms- folder (7) modify Build.rules so repbuild is passed -ba instead of -bb, so

the source rpm is generated (8) add SRPMS sub-folder inside build_files.tgz (9) modify make_dud so the source rpm is copied to the release/rpms-$release folder

- CQ 148348 - Increasing max_queue_depth in Linux Driver not honored above 30,000 - Bug fix in the queue depth calculation when queue depth xceeded the MaxReplyDescriptorPostQueueDepth. The algorithm was resizing the queues incorrectly.
- CQ 147144 - Linux Phase 8: PCIe Error Handling Fix's: (1) Reject all request generated internal inside the driver as well as request arriving from the scsi mid layer when PCIe EEH is active. A per adapter flag called pci_error_recovery is added, this flag is checked thru out the driver prior to sending FW generated requests. (2) Don't need to call the pci_driver->remove directly from the PCIe callbacks as this is already handled from PCIe EEH code. In its place the the watchdog timer is shutdown, and all pending IO is flushed back. (3) Save and restore pci state across PCIe EEH handling.
- CQ 147145 - the "internal device reset complete" event is not supported for older firmware prior to MPI Rev K (which is Phase 3) - We added a check in the driver so the "internal device reset" event is ignored for older firmware. When ignored, the tm_busy flag doesn't get set nor cleared. Without this fix, IO queues would be frozen indefinitely after the "internal device reset" event, as the "complete" event never sent to clear the flag.
-

Major Changes For Version 07.255.04.00-1
Release Date: 08/04/2010

- CQ 141553 - When zoning end devices, the driver is not sending device removal handshake algorithm to firmware. The results controller firmware not sending sas topology add events the next time the device is added. The fix is the driver should be doing the device removal handshake even though the PHYSTATUS_VACANT bit is set in the PhyStatus of the event data. The current design is avoiding the handshake when the VACANT bit is set in the phy status.

Major Changes For Version 07.255.03.00-2
Release Date: 07/15/2010

- CQ 147043: Gen2 Linux: The sles11 KMPs contain compiled drivers for SLES11 SP1 kernel: Modified make_rpm script so softlinks for each kernel flavor are created prior to calling rpmbuild.

Major Changes For Version 07.255.03.00-1
Release Date: 06/10/2010

- CQ 142911 - IO's to the target mode luns stops on port reset to the Target mode controller - When the target mode driver is reset, the links will go

down then up. The initiator will process events to block and unblock IO for the link status changes. Prior to unblocking IO, the driver will send a Test Unit Ready, followed by start unit if required. The response returned to initiator for TURs is MPI2_SCSI_STATE_TERMINATED. Due to bug in initiator driver, the MPI2_SCSI_STATE_TERMINATED is treated as a success. The driver will subsequently next ask for serial number which will fail, then the driver will remove the device, and add it back later. To fix this issue, the driver will need to solve the processing the scsi state so non-zero value is returned with DEVICE_RETRY instead of DEVICE_READY. When DEVICE_RETRY is returned, the TURs will be retried several times. When scsi state transitions back to non-zero value, the serial number check will work, and the device will not be deleted.

- CQ 136939 - Bad request seen after many hours in reset expander test The firmware engineer reported that the driver was sending a SCSI_IO with the device handle set to 0xFFFF. The driver would only set the device handle to 0xFFFF after having received the device deletion event. Though nearly impossible that IO would be sent to end device after having received a device deletion, there is two cases where an small window is open to this. (1) IO Sent to the normal IO queue command function: There is a sanity check for the 0xFFFF device handle at the top of function, and however it is possible that the device handle could change by the time MPI function was filled out with the device handle. (2) Internal generated SCSI_IO: Typically this is only called when devices are added. Highly unlikely a device add and device delete are going on simultaneously. However there is no check in the routine for a device handle 0xFFFF. The fix will address both cases to include a sanity check on device handle of 0xFFFF. It will be impossible that 0xFFFF device handle will ever be sent to firmware.
- CQ 140365 - reopen/change_queue_depth callback ABI changed: In _scsih_slave_configure, when calling _scsih_change_queue_depth use qdepth instead of sdev_queue_depth in the 2nd parameter. The queue depth calculated in the slave function should be passed.

Major Changes For Version 07.255.02.00-1
Release Date: 05/21/2010

- CQ 140398 - adding SLES11 SP1 support in builds kits
- CQ 140429 - fix oops loading driver when there is direct attached SEP device: The driver set max phys count to the value reported in sas iounit page zero. However this page doesn't take into account additional virtual phys. When sas topology event arrives, the phy count is larger than expected, and the driver accesses memory array beyond the end of allocated space, then oops. Manufacturing page 8 contains the info on direct attached phys. For this fix will making sure that sas topology event is not processing phys greater than the expected phy count.
- CQ 140365 - change_queue_depth callback ABI changed: The change_queue_depth callback changed where there is now an additional

parameter called reason, with SCSI_QDEPTH_DEFAULT, SCSI_QDEPTH_QFULL, and SCSI_QDEPTH_RAMP_UP codes. This enhancements add support for these reason codes for 2.6.32 or greater kernels.

- CQ 140293 - remove support for MPI2_EVENT_TASK_SET_FULL: This event is obsoleted in Phase 7, so this processing of this event needs to be removed from the driver. The controller firmware is going to handle TASK_SET_FULL, the driver doesn't need to do anything. Even though we are removing the EVENT handling, the behavior has not changed between driver versions because firmware will still be handling queue throttling, and retrying of commands when the target device queues are full.
- CQ 140292 - MPI2 Rev O (2.0.7) and v2.00.16 header files (for Phase 7.0)

Major Changes For Version 07.255.01.00-1
Release Date: 05/07/2010

- CQ 134854 - Fix panic in the driver interrupt routine when controller firmware returns an invalid system message id (smid). The oops is occurring because mpt_callbacks pointer is referenced to either NULL or invalid virtual address. This is due to cb_idx set incorrectly from routine _base_get_cb_idx. The cb_idx was set incorrectly because there is no check to make sure smid is less than maximum anticipated smid. To fix this issue, we add a check in _base_get_cb_idx to make sure smid is not greater than ioc->hba_queue_depth. In addition, a similar check was added to make sure the reply address was less than the largest anticipated address. The underlying issue is in firmware bug. The firmware bug was eventually solved in phase 5, however it was suggested by a firmware engineer that the driver have these additional safety net.
- CQ 120445 - Support DIF Type 2 Protection for 2.6.33 kernel: Adding DIF Type 2 protection support, as well as turning on 32 byte cdb's, and setting the cdb length for > 16 byte in the SCSI_IO->control parameter.
- CQ 121492 - ability to override/set the ReportDeviceMissingDelay and IODeviceMissingDelay from driver: Add new command line option missing_delay, this is an array, where the first element is the device missing delay, and the second element is io missing delay. The driver will program sas iounit page 1 with the new setting when the driver loads. This is programmed to the current and persistent configuration page so this takes immediately, as will be sticky across host reboots.
- CQ 123622: create a pool of chain buffers, instead of dedicated per IO: This enhancement is to address memory allocation failure when asking for more than 2300 IOs per host. There is just not enough contiguous DMA physical memory to make one single allocation to hold both message frames and chain buffers when asking for more than 2300 request. In order to address this problem we will have to allocate memory for each

chain buffer in a separate individual memory allocation, placing each chain element of 128 bytes onto a pool of available chains, which can be shared among all request.

- CQ 135828 - switch swap doesn't work when device missing delay is enabled. (1) add support to individually add and remove phys to and from existing ports. This replaces the routine _transport_delete_duplicate_port. (2) _scsih_sas_host_refresh - was modified to change the link rate from zero to 1.5 GB rate when the firmware reports there is an attached device with zero link. (3) add new function mpt2sas_device_remove, this is wrapper function deletes some redundant code through out driver by combining into one subroutine (4) two subroutines were modified so the sas_device, raid_device, and port lists are traversed once when objects are deleted from the list. Previously it was looping back each time an object was deleted from the list.

Major Changes For Version 06.00.00.00-1
Release Date: 05/05/2010

- Bump Driver Version for GCA

Major Changes For Version 05.255.05.00-1
Release Date: 05/03/2010

- CQ 136658/CQ 133558 - Ph6 IT: IO errors while running Io's and Abort task every 7 secs: To recreate the issue, send IO to devices using /dev/sd device nodes, then simultaneously sending abort task thru the IOCTL interface. The problem is after some time, the IO stops being sent to target because the IO queues are are in frozen stuck state. The bug is the driver is due to not clearing the per device tm_busy flag following the Task Management request completion from the IOCTL path. When this flag is set, the IO queues are frozen. The reason the flag didn't get cleared is because the driver is referencing memory associated to the mpi request following the completion, when the memory had been reallocated for a new request. When the memory was reallocated, the driver didn't clear the flag because it was expecting a task management request, and the reallocated request was for SCSI_IO. To fix the problem the driver needs to have a cached backup copy of the original request.

Major Changes For Version 05.255.04.00-1
Release Date: 04/12/2010

- CQ 131980 - Add RHEL5.5 support: modify build kits so pre-built binaries are provided for 2.6.18-194 kernel.
- CQ 131614 - Sense data buffer not available for SCSI IO passthrough responses: (1) driver was not setting the sense data size prior to sending

SCSI_IO, resulting in the 0x31190000 loginfo. (2) The driver needs to copy the sense data to local buffer prior to releasing the request message frame. If not, the sense buffer gets overwritten by the next SCSI_IO request.

- CQ 131132 - Linux Driver-Kernel Panic while running sg utils during Enclosure/Drive removals: (1) The sg driver is sending an INQUIRY between the DELAY_NOT_RESPONDING and TARG_NOT_RESPONDING events. Between those events the device is in blocking state and INQUIRY is held up in the request_queue. The INQUIRY is never dispatched to the driver by the time the device is deleted. Eventually the sg driver completion routine is called, then oops because its accessing a stale file descriptor mapped to the device that was just deleted. The fix was to to give more time for the pending IO that was blocked in the request_queue to be dispatched to the driver; that way the sg completion routines are accessing valid file descriptors. The fix is to unblock devices when the driver first is notified of the TARG_NOT_RESPONDING event from interrupt context, instead from the delayed worktask context. Also, setting the device in SDEV_OFFLINE prevents the SYNCHRONIZE_CACHE being issued to the scsi mid-layer for devices having WCE enabled. (2) Unable to unload the driver after simultaneously sending sg_reset -h to driver and pulling devices while host reset is active. The reason is because there are open file handles. The fix is to give the sg driver control back before mpt2sas begins report missing devices.
- CQ 131975 - checkpatch and compile warning fixes.

Major Changes For Version 05.255.03.00-1
Release Date: 04/05/2010

- CQ 129932 - Incorrect checking of pci resource flag using PCI_BASE_ADDRESS_SPACE_IO

On a customer specific PowerPC platform, the driver failed to load due to incorrect mapping of PCI Memory Resources. The driver fix is to add additional sanity check to make sure the driver traverses pci resources which are IORESOURCE_MEM prior to calling ioremap.

- CQ 130042 - fix the incorrect scsi_dma_map error checking:

scsi_dma_map returns the number of sg lists actually used, zero if the sg lists is NULL, or -ENOMEM if the mapping failed. The driver was not expecting a return value of -ENOMEM, hence when -ENOMEM was returned, it would think there were a huge number of sg elements. The fix is to change the parameter "sges_left" to an integer (signed), and return error from _scsih_build_scatter_gather function when scsi_dma_map returned a negative value.

- CQ 130633 - Volume not deleted after host reset

  BUG DESCRIPTION:
  This test case involves creating two RAID1 volumes, then simultaneously issue host reset and pull all the drives associated to the 1st raid volume. The observed behavior is the physical drives are removed, however the volume remains. The expected behavior is the volume as well as physical drives should be removed from OS.

  BUG FIX:
  Add support in the post host reset device scan logic for raid volumes where the driver will have an additional check for responding raid volume where the status should be either online, optimal, or degraded. So for volumes that have a status of missing or failed, the driver will mark them for deletion.

- CQ 125138 - Add additional sas_address/phy messages in driver.

Adding additional messages to the error escalation callbacks which displays the wwid, sas address, handle, phy number, enclosure logical id, and slot. In the same eh callbacks, routines, the printks were converted to sdev_printks, which displays the bus target mapping. These additional modifications help better identify the device which is in recovery.

- CQ 125137 - Timeouts in mulit-pathing environment in SLES11.

BUG DESCRIPTION:
Setup SAS drive with two cables connected to controller. Load the driver with the mpt2sas_multipath command line option enabled. Start the device mapper driver stack. Send IO to /dev/dm-1. After some time has elapsed, then issue target reset to /dev/dm-1 using the tool `sg_reset -d /dev/dm-1`.

Bug Fix:
Add support in the function mpt2sas_scsih_check_tm_for_multipath so the driver processes LOGICAL_UNIT_RESET. Currently it was only supporting TARGET_RESET. So with this fix, the driver will be sending an ABRT_TASK_SET over to the other path, this will flush out all the command that were dropped to the floor due to either LOGICAL_UNIT_RESET or TARGET_RESET. Please note that under SLES10/RHEL5, the LUN_RESET is not support.

- CQ 116228 - HBA doesn't tell host about device when released from reset

Add support in the drivers unblock routines so the driver will retry _scsih_wait_for_device_to_become_ready function when the its return code is either RETRY or DEVICE_START_UNIT. For the case of ESG targets, it doesn't support START_UNIT with IMMED=1, so we need will to retry that command as well. The unblock routines are called either the case when the phy links are coming back before DMD timer expires. The unblock routine is also called from host reset when the device was previously blocked (to see if we can unblock them).

Add support for keeping device handles for devices that are in the process of being added to the topology. In the case of BUSY device, the driver will be retrying TUR's every one second. We need to maintain this device handles so we know the device is in progress of being added. We will handle a special test case for devices returning BUSY for a long period of time, then the device is removed and added again before the DMD timer expires. When the device is added back, there will not be a device add event, only event the driver will receive is phy links up event. With the new pending handles list, the driver will know that it was previously being added, and we will convert the links up event to a target add event.

- CQ 131059 - While sending IO to devices, an internal generated host reset results in driver returning DID_NO_CONNECT for all scsi commands.

BUG DESCRIPTION:
A change from CQ 129053 created this issue. We added support to set the deleted flag prior to device scan, then clear the flag for responding devices, leaving the deleted flag only set for missing devices. The problem is for internal generated host resets, IO queues are not blocked at scsi mid layer level. IO will be continued sent to driver, and driver will return SCSI_MLQUEUE_HOST_BUSY. The problem is the driver checks for the deleted flag before it checks for the controller being in reset, so there is a small window that the driver would be returning DID_NO_CONNECT for responding devices.

BUG FIX
Fix the queuecommand entry point so ioc->shost_recovery flag sanity check is given higher precedence then the device "deleted flag" check.

- CQ 131060 - discovery kicks off after port enable returns

Add support in driver so it will wait for discovery to complete before returning from the send_port_enable routine. There are some cases where firmware is doing discovery after port enable completes, when it does this, the driver will not have all the devices in the sas_device_init_list list prior to calling the sort routine for reporting boot devices to OS.

Major Changes For Version 05.255.02.00-1
Release Date: 03/29/2010

- CQ 129053 - Data corruption after drive pull during diag reset (CSET 129385)

  RECREATE:
  This issue surrounds issuing a host reset to the controller then at the same time a single drive is removed. Following host reset, the driver will send a port enable to firmware which results in a rediscovery of all devices. Sometimes a rediscovery of devices can result in device handle assignments changing, especially when a single device is removed.

  BUG DESCRIPTION:
  The bug is the device driver is scanning for devices after host reset processing completes. Since we are refreshing the handles after reset, there is a window where IO could be going to the old device handle.

  BUG FIX:
  The changes in this patch is as follows:

  (1) The driver will be scanning for devices before the host processing completes, and appropriately setting device handles to their new assignments. The three functions _scsih_mark_responding_xxx have been moved from the hotplug worktask to to host reset context.

  (2) A new function was added to set all devices to the deleted state, then clear the flag during the device scan for responding devices. At conclusion of the device scan, only missing devices will be left with the deleted flag set. With the deleted flag set, the driver will return DID_NO_CONNECT for all scsi commands during such time the hotplug worktask was activated to report all missing devices to the scsi mid layer.

- CQ 125906 - Mpt2sas and blocked devices (CSET 129830)

  Bug Description:
  If error recovery is occurring at the same time when a device is blocked, the escalation will alway escalate to host reset. The reason for escalating is due to TURs sent by scsi mid layer function scsi_eh_tur after each Task Management request. Those TURs are failing because the driver is returning SCSI_MLQUEUE_DEVICE_BUSY from the queuecommand callback.

  Bug Fix:
  The change is for the driver to return DID_OK for the TURs sent during

error recovery while the device is blocked. From the scsi mid layer scsi_unjam_host function, when TURs succeed, the eh_scsi_cmd is going to be moved from eh_work_q to eh_done_q. At the end of error recovery, all the scsi_cmnds on the eh_done_q, are put back on the request queue from the scsi_eh_flush_done_q function. Those commands are retried and sent to target (hopefully unblocked by that time). The point is there is no reason to escalate error recovery for blocked devices.

- CQ 125906 - Mpt2sas and blocked devices (CSET 129830)

  Bug Description:
  A misbehaving drive where links are fluctuating sometimes result in device left infinitely in blocked state. When a device is left in a blocked state, no subsequent request can get through to the device, which make it appear as though the application is hung.

  Bug Fix:
  When the links come up, the device driver sends a series of scsi commands to the device in order to check for the responsiveness of the device. If the series of commands succeed, then the driver will unblock. However if they fail, the driver was leaving the device blocked. The change in behavior of this fix is set the delete flag, then unblock the device. Any subsequent IO will be returned with DID_NO_CONNECT. In addition, the driver changes the state to SDEV_OFFLINE.

- Red Hat Bugzilla 567965 - add missing initialization: In the driver mpt2sas_base_attach subroutine, we need to add support to return the proper error code when there are memory allocation failures, e.g. returning -ENOMEM.

- CQ 125321 (child activity to CQ 118104) : Add old behavior into driver for RAID configuration when removing devices: The new driver behavior will only send target reset to PD when there is a single drive pull from a volume. This change occurred in CQ 118104. The older driver would be sending target reset to volume when there was a single drive pull from volume. The fix in the CQ is to send target reset to volume when there is a single drive pull, wait for the reply, then send target resets to the PDs. We want to add this behavior back so the driver will behave the same for older versions of firmware.

Major Changes For Version 05.255.01.00-1
Release Date: 02/20/2010

- CQ 118104 - Fault 0x600F during NEM pull: REPRODUCE STEPS: To reproduce this issue, you need to have multiple expanders between the

hard disks and host controller. Create a RAID volume, then start running IO stress application for several minutes. Then pull the cable between the host controller and top level expander, this will result in fault. BUG DESCRIPTION: The fault is a result of the driver receiving the top level expander removal event prior to all the individual PD removal events, hence the driver is breaking down all the PDs in advanced to the actual PD UNHIDE event. In addition. the fault is a result of the driver sending multiple Target Resets to the same volume handle for each individual PD removal. BUG FIX DESCRIPTION: To fix this issue, the entire PD device handshake protocol has to be moved to interrupt context so the breakdown occurs immediately after the actual UNHIDE event arrives. The driver will only issue one Target Reset to the volume handle, occurring after the FAILED or MISSING volume status event arrives from interrupt context. For the PD UNHIDE event, the driver will issue target resets to the PD handles, followed by OP_REMOVE. The driver will set the "deleted" flag during interrupt context. A "pd_handle" bitmask was introduced so the driver has a list of known pds during entire life of the PD; this replaces the "hidden_raid_component" flag handle in the sas_device object. Each bit in the bitmask represents a device handle. The bit in the bitmask would be toggled ON/OFF when the HIDE/UNHIDE events arrive; also this pd_handle bitmask would be refreshed across host resets.

- CQ 119210 Experienced a node crash early in the boot process while using the mpt2sas driver: BUG DESCRIPTION: Fix a oops in _scsih_sas_device_remove. The driver was attempting to delete a object from the sas_device link list when the object was not present. BUG FIX DESCRIPTION: Add fix in _scsih_sas_device_remove, where it checks for presence of sas_device object prior to deleting it from the link list.
- CQ 119208 Tie a loginfo message to a specific PHY: Add support to display additional debug info for SCSI_IO and RAID_SCSI_IO_PASSTHROUGH sent from the normal entry queued entry point, as well as internal generated commands, and IOCTLS. The additional debug info included the phy number, as well as the sas address, enclosure logical id, and slot number. This debug info has to be enabled thru the logging_level command line option, by default this will not be displayed.
- CQ 123883 - add expander phy control support: Add support to send link resets, hard resets, enable/disable phys, and changing link rates for for expanders. This will be exported to attributes within the sas transport layer. A new wrapper function was added for sending SMP passthru to expanders for phy control.
- CQ 123882 - add expander phy counter support: Add support to retrieve the invalid_dword_count,running_disparity_error_count, loss_of_dword_sync_count, and phy_reset_problem_count for expanders. This will be exported to attributes within the sas transport layer. A new wrapper function was added for sending SMP passthru to retrieve the expander phy error log.

- CQ 119206 When a drive returns an 04/19/01 (sense/asc/ascq) at boot so the HBA driver will still allow the drive to be seen by the OS: Add support in hot plug add code. The driver will allow reporting an end device to scsi mid layer when a TUR returns 04/19/01.
- CQ 123655 - Converting KERN_DEBUG over to KERN_INFO: This is so when debuging issues, the critical messages are not missing from the logs. By default, RedHat/SuSe ship with KERN_DEBUG turned off.
- CQ 95473 - add support so the diag ring buffer can be pulled via sysfs: Added three new shost attributes: host_trace_buffer, host_trace_buffer_enable, and host_trace_buffer_size. The host_trace_buffer_enable attribute is used to either post or release the trace buffers. The host_trace_buffer_size attribute contains the size of the trace buffer. The host_trace_buffer attribute contains a maximum 4KB window of the buffer. In order to read the entire host buffer, you will need to write the offset to host_trace_buffer prior to reading it. There is a script provided into the source code kit which will release the host buffer, then write the entire host buffer contents to a file. In addition to this enhancement, we moved the automatic posting of host buffers at driver load time to be called prior to port_enable, instead of after. That way discovery is available in the host buffer.
- CQ 123562 - adding Phase 6 MPI headers version N
- CQ 123538 : add counter for ioc_reset: Add a new sysfs shost attribute called ioc_reset_count. This will keep count of host resets (both diagnostic and message unit).
- CQ 123537 : Drive S/N check for SATA drives: Add support to obtain the serial number for all devices, it will stored in the sas_device object at detection time. The serial number is in INQUIRY VPD Page 0x80. Then the serial number will be requested again when links are coming back up after having been lost. If there is a serial number mismatch for matching sas address, the driver will delete the old device, then add the new.
- CQ 123536 : staged device discovery: Added command line option called disable_discovery. When enabled on the command line, the driver will not send a port_enable when loaded for the first time. If port_enable is not called, then there is no discovery of devices, as well as the sas topology. Then later if one desires to invoke discovery, then they will need to issue a diagnostic reset. A diagnostic reset can be issued various ways, either using an application like lsiutil, or via sysfs.
- CQ 114298 - Node crashed by performing device resets in parallel with bus/host resets : The driver had an oops is because a host reset was sent by scsi mid layer while there was already an host reset active, either issued via IOCTL interface or internally, like a config page timeout. Since there was a host reset active, the driver would return a FAILED response to the scsi mid layer. Then the scsi midlayer would begun cleaning up all the pending scsi commands memory blobs before the driver called scsih_flush_running_cmds, hence the scsi command request_buffer would be set to NULL. Then when the driver called pci_unmap_sg, the

scsi cmd request_buffer was NULL, and we oops. The solution is make sure pending host resets will wait for the active host reset to complete before returning control back up the call stack.

- CQ 118325 - Enclosure_identifier not being returned by mpt2sas: The driver exports callback function to the sas transport layer for obtaining the enclosure logical id. This function is called _transport_get_enclosure_identifier. The driver bug is due to searching the wrong list for the enclosure_identifier. The driver should be searching the sas device list instead of enclosure list. The sas address that is passed to the driver is for the end device, not enclosure.

Major Changes For Version 05.00.00.00-1
Release Date: 02/09/2010

- Create GCA release.

Major Changes For Version 04.255.05.00-1
Release Date: 01/21/2010

General Changes

- CQ 117637 - bug fix in the handling of the internal device reset event The reason code check in scsih_sas_device_status_change_event never evaluates as true for internal device reset, hence driver never quiesce s IO when firmware is sending a device reset. The fix is to change the evaluate to:

  ```
  if (event_data->ReasonCode !=
     MPI2_EVENT_SAS_DEV_STAT_RC_INTERNAL_DEVICE_RESET &&
    event_data->ReasonCode !=
     MPI2_EVENT_SAS_DEV_STAT_RC_CMP_INTERNAL_DEV_RESET)
          return;
  ```

- CQ 114786 - fix NULL pointer dereference in mpt2sas_base_hard_reset_handler: The completion "ioc->shost_recovery_done" was not initialized prior to calling complete() near the end of the function call. Typically it would of been initialized from the hotplug work task function _mpt2sas_fw_work, however in this particular test case, the hard reset function completed to the end of the function before the work task got \activated, hence the null pointer dereference. To fix this issue, we need to initialize "ioc->shost_recovery_done" at driver load time.
- CQ 118959 - Updated copyright to year 2010
- CQ 119002 - MPI2 proper descriptor usage for RAID_SCSI_IO_PASSTHROUGH: Driver needs to be sending the default descriptor for RAID Passthru, currently its sending SCSI_IO descriptor.

This was clarified in the MPI specification under CQ 108546. The driver changes were in three places: queue_command, internal request, and the ioctl mpi_passthru's.

- CQ 119225 - _scsih_determine_disposition - in switch, missing break for MPI2_IOCSTATUS_SCSI_IOC_TERMINATED: The driver is missing a break inside the switch/case for ioc_status. Since the break is missing, the function is returning the incorrect disposition DEVICE_READY for the case when TURs are sent during cable pull. Also added support in the same function so devices returning sense key NOT_READY, with ASC=0x3E, meaning device is self configuration, so the START_UNIT is sent later from the calling function.
- Reopen – 114752: Need to fix a check in scsih_tm_tr_mp_send where it is making sure the controller is not in reset or driver unloading. The bug fix was driver should of been checking status on the alternate controller instead of current controller.

Major Changes For Version 04.255.04.00-1
Release Date: 12/23/2009

General Changes

- CQ 113882 - incorrect timestamp on 32 bit platforms: The upper 32 bit of the timestamp was getting truncated when converting seconds to milliseconds, which was due to the variable being long. To fix the problem, the variable needs to be u64. Also the microseconds conversion to milliseconds was incorrect; it should be divide by 1000 instead of divide by 8.
- CQ 113732 (CSET 114490) - setting queue depth for meteor: The driver failed to load due to a very large memory allocation failure when creating the ioc->scsi_lookup. To fix this issue, we need to use the get_free_pages API. Also, the ioc->chain_depth need to be changed from a 16bit to 32bit variable because the number of chains will exceed 64k when the queue depth is large.
- CQ 113748 - SLES11 RC6 Kernel dump upon cable removal: There is a bug in the SLES 11 kernel enclosure services driver called enclosure.ko. from the function enclosure_unregister. The enclosure driver will hang when removing a cable attached to HP or Xyratex enclosures. The reason for the hang is the enclosure driver didn't properly detect the SEP device during device scan, thus when the device was deleted, it tried removing objects from a link list which was not populated. Apparently this issue was solved later in SLES11 SP1. The fix in this release is to blacklist the enclosure and ses device drivers so they are not loaded, hence avoiding the hang condition. This fix is handled from the build kits (duds, standard rpms, and dkms), as well as the driver compile script.

- CQ 114752 - linux driver fault 1500: The fault 1500 is a result of the driver sending command to firmware during message unit reset. This issue is reproduced by rebooting the system, then at the same time pulling cable to an enclosure full of drives. The problem was due to hotplug work threads being still active when the shutdown routine was called. To fix this problem the driver will set the flag "ioc->remove_host" from the shutdown routine, then also break down the hotplug work threads, and any pending work. Also added were several sanity check for the flag "ioc->remove_host".

Major Changes For Version 04.255.03.00-1
Release Date: 12/15/2009

General Changes

- CQ 111950 - debug logging not getting compiled in to the driver for SLES10 SP3 kernels. In SLES10 SP3, the messages for debugging is not compiled into the driver even though the define for CONFIG_SCSI_MPT2SAS_LOGGING is enabled in the driver Makefile. We don't see this problem for earlier SLES10 service packs or SLES11. The change is to define this from linux-compatibility.h, so this is always enabled for internal builds. For upstream kernels, this should be enabled by end user via Kconfig.
- CQ 112841 - sata devices not discovered: A change in previous CQ 110388 changed this behavior. This change caused sata devices with access flags bits set to SATA_NEEDS_INITIALIZATION to fail. Actually this is not a failure since sata devices will return this bit set prior to the first a scsi command sent. So to fix this, the driver will not treat NEEDS_INIT as failure. In addition to this fix, the driver will now display message to describe the the access flags when bits are set, so the end user can better understand failures.
- CQ 113305 - hotplug/diag reset: This CQ created to clean up code related to test cases where devices are added or removed while diagnostic reset is active. There is one bug fix which was created when solving another issue; CQ 107285. Basically this bug fix surrounds the problem where devices are not being removed from the operating system when diagnostic reset is sent. In addition to this fix, we have enhanced the algorithm surrounding the scanning and deletion of missing devices following host reset. What we did is moved this code to the hotplug work threads from diag reset context and watchdog timer. The purpose of moving this code is to prevent race condition when driver was deleting and adding the same device at the same time. During this race, it is possible to hit an oops in the sas transport layer when phys are attached to two separate ports at the same time.
- CQ 111140 - driver fails to compile when MULTIPATH flag is disable: The parameter tm_tr_mp_cb_idx was added in previous bug fix. We should

place this parameter be under the "#ifdef MPT2SAS_MULTIPATH" in order to fix the compile problem. This parameter was added when we were separating the multipath target reset handling from normal driver/firmware device removal protocol handshake.
- CQ 101922 - ioctl returns success on timeouts: The driver was modified to return -ENODATA when there is a timeout via ioctl path.

Major Changes For Version 04.255.02.00-1
Release Date: 11/20/2009

General Changes

- CQ 108971 - Fix endian issues in target mode driver: Modified driver so the subroutine mpt2sas_base_put_smid_target_assist is not compiled in for non target mode sources.
- CQ 107285 - external host not connecting after controller reboot: The reported problem is : devices are not coming back after having the cable disconnected then reconnected. The problem is because the driver/firmware device removal handshake is failing. Due to this failure, the controller firmware is not sending out device add events when the target is reconnected. This is root caused to a race in the driver/firmware device removal algorithm. There is duplicate code in both interrupt and user context; where target reset is being issue from user context path while sas_iounit_control(OP_REMOVE) is being sent from interrupt context. An active target_reset will fail the OP_REMOVE. To fix this problem, the duplicate code has been removed from user context path.
- CQ 104522 - host reset hangs when pulling a drive while running IO stress test: The design of the driver is to remove non-responding devices following host reset. When devices are removed, the scsi mid layer will send SYCN_CACHE scsi passthru if WCE is enabled. This can't be done during host reset because the scsi mid layer puts controller into SHOST_RECOVERY state, which freezes all IO to devices, hence deadlock occurs. The fix this problem, we need process device removals outside the host reset context. This code was moved to the watchdog subroutine.
- CQ 110388 - too many report luns/turs: The report_luns/tur logic is in the driver used to check responsiveness of devices prior to reporting to scsi mid layer, and/or unblocking. Apparently the driver currently is sending too many turs unnecessarily and redundantly. This driver patch cleans up the code so report luns is only issued when device are first reported; then report_luns will be skipped since the lun number is available. Following host reset, we will only send turs/spin_ups for devices in blocking state; e.g. its not required to send turs for devices that were already responding

prior to the reset. Also for wide ports, we will only send turs for one phy, instead of every phy. Another fix is to refresh the device handles following phy link status change.

- CQ 110306 - ppc64 sas2flsh failure investigation: On ppc64, an 32bit application was failing due to data buffers not being copied properly from user to kernel memory. The problem due to improper conversion of 32 to 64 bit pointers. The fix is to use compat_ptr to setup the pointer compatibility in the routine _ctl_compat_mpt_command.

Major Changes For Version 04.255.01.00-1
Release Date: 11/06/2009

General Changes

- CQ93843 – DKMS Kit, kdump ramdisk was not getting updated with new driver version: (1) for SLES10, updated rpm spec will create a symbolic link from vmlinux to vmlinuz kdump kernel – that way the kdump ramdisk will get updated with newer driver version. (2) for SLES11 and RHEL5, updated rpm spec file will call mkdumprd, which will update kdump ramdisk image with new driver version (3) remove logic in kit which modifies kernel configuration when compiling the driver, this includes removing the script dkms_pre_build from the kit.
- CQ 107203 - remove scripts from build kits that are modifying kernel config: A customer has suggested its not a good idea modifying the kernel configuration when installing RPMS, therefore we Modified the driver Makefile with “obj-m += mpt2sas.o”
- CQ 105155 - improve displayed message when setting new fwfault_debug
- (1) change the string logging_level to fwfault_debug, (2) set ioc->fwfault_debug to the command line option mpt2sas_fwfault_debug setting at driver load time.
- CQ 107107 - Use resource_size_t to define the type: resource for the system interface register set. The existing implementation was using "unsigned long" which would be 32 bit in 32 bit OS. A customer reported that his 32 bit OS is using 64 bit physical address space for the system interface register set, therefore we need to shift to using resource_size_t which takes care of physical address space.
- CQ 104563 - little endian fix's
- CQ 106191 - adding new phase 5 headers - revision M
- CQ 94542 - add support to enable/disable phys, and change link rates: Added new callbacks phy_enable and set_phy_speed in the mpt2sas_transport_functions template. This will allow end user to enable/disable phys and change links rates using the SysFS interface.
- CQ 104554 - adding raid transport layer support: Adding support for raid transport layer. This will provide some sysfs attributes containing raid level, state, and resync rate.

- CQ 107202 - raid transport support in SLES11 duds: (1) raid_class.ko needs to be included on the DUD. (2) add raid_class.ko in the file called module.order
- CQ 104243 - increase the scatter gather element size to 256: This provides capability to send larger data transfers. This is only present in SuSE distributions. By default the driver will be using 128 sgl elements. In order to use 256, one will need to use the command line option as follows:

  example: # insmod mpt2sas.ko max_sgl_entries=256
- CQ 70095 - We have added support in the driver to support EEH and PCIe Advanced Error Recovery. This involves adding new pci_error_handler interface for recovering the controller from PCI Bus errors, such as SERR and PERR. Some tools are available for simulating PCI errors in order to validate this interface: http://www.kernel.org/pub/linux/utils/pci/aer-inject
- CQ 98428 - add mpt_sdev_queu_depth MOD_PARAM to mpt2sas driver: Add new command line parmeter sdev_queue_depth. This will globally set the queue depth for all SAS devices at once(ignoring SATA and volume). This parameter can be set at driver load time, or on the fly using sysfs interface.

  Example loading driver:
  # insmod mpt2sas.ko sdev_queue_depth=500

  Example setting on the fly:
  # echo 500 > /sys/module/mpt2sas/parameters/sdev_queue_depth

Major Changes For Version 04.00.01.00-1
Release Date: 10/19/2009

General Changes

- CQ 102826 - adding MODULES_CONF_ALIAS_TYPE to dkms.conf: This is a request from the customer to add ALIAS_TYPE string to the dkms.conf file specifing this driver is a scsi_hostadapter. Note, this is part of the DKMS packing build kit.
- CQ 102810 - system halts during jammer scripts and medusa stress: There is an oops when a host reset is called while the driver hotplug work threads are active. This oops root caused to the routine _scsih_remove_device. This function was re-entered while in the process of deleting a device from work task context, then at the same time from host reset context when invoking a topology rescan. Since the routine was active at the same time from two different contexts, a panic resulted due to

sas_device deleted twice from the link list ioc->sas_device_list. The fix for this race condition is to immediately remove the sas_device from the link list at the top of the function. An additional fix was added to make sure all pending work was canceled across diag_reset; the fix was to add the cancel_pending_work flag from the fw_event_work structure, and then to set the flag during host reset, check the flag later from work threads context.

Major Changes For Version 04.00.00.00-1
Release Date: 10/12/2009

General Changes

- bump driver version for GCA
- CQ 101922 - ioctl returns success on timeouts: The driver was modified to return -ENODATA when there is a timeout via ioctl path.

Major Changes For Version 03.255.04.00-1
Release Date: 09/24/2009

General Changes

- CQ 99635 - adding SLES10 SP3 support in driver build kits
- CQ 99494 - phase 4 linux driver rpm install of dkms results in errors: The Phase 4 dkms is built under SLES11. The compression type for rpms changed from bzip2 to lzma, between SLES10 and SLES11. The lzma compression type is not supported in SLES10, hence the failure. In order to fix this problem, we need to force the dksm rpms built under SLES11 to be compressed with bzip2. This is done by adding "%define _binary_payload w9.bzdio" in the rpm spec file.
- CQ 98984 - erasing and flashing the new expander firmware hangs the system: Added new function which will make sure when new port is added, that its not claiming the same phy resources already in use by another port. If it does, then it will delete the other port before adding the new port. If we don't release the claimed phy resources, the sas transport layer will hang from the BUG in sas_port_add_phy.
- CQ 98420 - Power PC - unloading driver hangs when there is a volume The driver hangs when doing `rmmod mpt2sas` if there are any IR volumes present. The hang is due the scsi midlayer trying to access the IR volumes after the driver releases controller resources. Perhaps when scsi_remove_host is called, the scsi mid layer is sending some request. This doesn't occur for bare drives because the driver is already reporting those drives deleted prior to calling mpt2sas_base_detach. To solve this issue, we need to delete the volumes as well.
- CQ 98589 - snowmass - io stress and random TM causes STAF script to lockup: Steps to reproduce the issue: while running IO to multiple device, send random task management request via the mptctl IOCTL interface.

Eventually the console running the task management request stops responding, meaning everything else is still responding from the other consoles The message logs show that there are scsi mid layer error handling escalation from various hard disk, meanwhile the console running the test case has hung. Analyzing all the stack traces using sysrq reveal that the perl script is in a mutex deadlock. To fix the issue, we have removed all the mutex's for ioc->tm_cmds.mutex, then created one single mutex inside the function mpt2sas_scsih_issue_tm. This is the single function used when sending task management. Also the sanity checks required for scsi mid layer escalation were moved to inside the same function because these checks need to be done while the mutex is held. The ioc->tm_cmds.mutex inside the IOCTL branch is really not required since there is another mutex in this code called for ctl_cmds handling this sync.

- CQ 99282 - mpt2sas: fix compile error when disabling MPT2SAS_MULTIPATH: The driver doesn't compile when multipath support is disabled. This is due to delayed_tr_list not being complied into the MPT2SAS_ADAPTER struct.
- CQ 99275 - mpt2sas: fix compile warnings: The flags should be unsigned long, instead of unsigned long long.

Major Changes For Version 03.255.03.00-1
Release Date: 08/31/2009

General Changes

- packaging updated with RHEL5.4 support – Linux kernel 2.6.18-164.el5
- CQ 94703 - sles11 driver update disks are missing: The sles10 version of the make_dud script for generating the deliverable was mistakenly checked in under sles11 kit.
- CQ 95191 - freeze the sdev IO queue when firmware sends internal device reset: When receiving the MPI2_EVENT_SAS_DEV_STAT_RC_INTERNAL_DEVICE_RESET event, the driver will set the tm_busy flag in the sdev private host data, when tm_busy flag is set, the driver will return SCSI_MLQUEUE_DEVICE_BUSY, effectively freezing the IO to the device. The tm_busy flag is cleared with the MPI2_EVENT_SAS_DEV_STAT_RC_CMP_INTERNAL_DEV_RESET event.
- CQ 97548 – DID_TRANSPORT_DISRUPTED: A request to return DID_TRANSPORT_DISRUPTED instead of DID_BUS_BUSY when there is nexus loss. This will only effect SLES11, as well as RHEL5.4 and SLES10 SP3. For older kernels, the driver will still return DID_BUS_BUSY. What this will allow is the multi-path driver to delay the fail over process. They would like the path to keep up as long as the

device missing delay, which is 10 seconds. With DID_BUS_BUSY, the path fails over immediately.

- CQ 97392 - add a bad disk to the OS: Made an exception for medium errors of ASC=0x31. The test unit ready will allow the drive to continue on through the device add process.
- CQ 97420 - PPC (power pc) endian bug fix's: (1) EEDP(End to End data protection) was not working. This was due to not setting EEDP BlockSize and Flags to little endian format in the message frame. (2) Some expander sysfs attributes were not getting set properly. The sas format was not getting set due to endian issues with sas_format field in the struct rep_manu_reply. Since sas_format was not set properly, the component_vendor_id, component_revision_id, and component_id were not set. (3) In _transport_smp_handler: we don't need to convert the smid from little endian to cpu prior to calling mpt2sas_base_free_smid, because its already in cpu format. (4) Some loginfos and ioc status were not converted from little endian to cpu.
- CQ 96109 - mpt2sas_base_get_sense_buffer_dma should be returning little endian: From the function mpt2sas_base_get_sense_buffer_dma add support to call cpu_to_le32 when calculating the physical dma address. This will properly handle endianess on big endian systems. The return value of this function was changed from dma_addr_t to __le32. Remove the typecasting of u32 when setting the SenseBufferLowAddress, since its already in __le32 format.
- CQ 96901 - fix compile warnings and errors on PPC: (1) fix compile error in _transport_smp_handler. This was a result of renaming data_len to resid_len, see CQ 94747. Apparently the change to struct request occurred in the 2.6.31. This compile error will only be seen in sles11, as the smp portal is only enabled since the 2.6.24 Linux kernel. (2) fix compile warnings resulting from displaying of the sas address in some debug prints. The fix was to typecast the sas address with (unsigned long long).
- CQ 95500 - cable pull testing: Its observed that the OS was sending request to the driver after it had been put into blocking state, so the driver was modified to return SCSI_MLQUEUE_DEVICE_BUSY. The parameter spin_up_drive was renamed to wait_for_device in the function _scsih_ublock_io_device, to better classify the meaning of this field. It really means that we need to wait for the device to be ready to accept scsi commands before we put it back into running state.
- CQ 95459 - mpt2sas_fwfault_debug - make this parameter set per controller instance: (1) created ioc->fwfault_debug parameter - which is per controller (2) created new function _scsih_set_fwfault_debug, which will globally set each instance of ioc->fwfault_debug (3) renamed the shost sysfs attribute from mpt2sas_fwfault_debug to fwfault_debug (4) modify the sysfs callbacks to read and write the attribute fwfault_debug individually for each controller instead of globally.

- CQ 95433 - making sysfs attribute task_management more robust: (1) Add support to freeze the scsi host IO queue while sending these TM's; this replaces the code that was freezing each device queue individually. (2) Add "out of hi-priority request" message when there are no available message frames. (3) Add support for going to sleep when there are no available message frames, then waking up later when frames come available. This should allow all the task management request to be sent when there is a shortage of free request message frames. (4) Add support to exit early when controller goes into fault state. (5) Add spin locks in the target reset loop when traversing the sas_device and raid_device link list (6) Add statistics support with a message that displays the number of task management request sent and the number of IOs that were terminated. (7) Add message to display the smid in the task management request and completion. For task abort it will be the smid that is being aborted, instead of the task management smid.
- CQ 95253 - fix spelling mistakes: Correct spelling mistake where functions and comments are using "clt" instead of "ctl"; mostly found in mpt2sas_ctl.c sources.
- CQ 95190 - Put controller in ready state when rebooting: Add support in the shutdown callback handler to put the controller in controller in READY state. This support is now going to be in both the driver unload and reboot context.
- CQ 95188 - Driver is not sending MUR at driver load time when EVENT_REPLAY capability bit is set The driver needs to retrieve the ioc facts prior to putting the controller into READY state. The current design is calling ioc facts after putting the controller into READY state, which means the driver is sending a diag reset instead of message unit reset because the capability information is not yet available.
- CQ 94963 - Phase 4.0 Linux Driver Unknown status displayed in dmesg: (1) for the MPI2_EVENT_SAS_TOPOLOGY_CHANGE_LIST event when the ExpStatus is set to zero, display "responding" instead of "unknown status". This actually fixed in CQ 94517 (2) for the MPI2_EVENT_IR_OPERATION_STATUS event, add support to print "background init" or "make data consistent" for debugging purposes. If the RAIDOperation is set to a value not defined, then then don't print anything (3) for the MPI2_EVENT_SAS_DEVICE_STATUS_CHANGE event, add support to print "expander reduced functionality" and "expander reduced functionality complete", which are new events.
- CQ 94517 - add IR support for the sysfs attribute task_management: The following changes were made in the contents of sysfs attribute task_management (1) remove support for sending task aborts to IR volumes and hidden raid components (2) add support to send target reset to IR volumes (3) remove support for sending target resets to hidden raid components (4) remove support for sending lun reset and abort task set hidden raid components (5) remove support for sending abort task to hidden raid components (6) target resets are sending link change rate

events with no link rate change -> thus said the driver was modified so when there is no link rate change, we don't need to call mpt2sas_transport_update_links nor _scsih_ublock_io_device. (7) There were changes made in _scsih_sas_topology_change_event_debug to change the debug strings so they are more clear. Also the link rate change information was added to display the new and previous link rate.

- CQ 94747 - add changes required by kernel.org: scsi_internal_device_block - scsi lld should use the block/unblock common API for blocking devices instead of setting the sdev state directly. The devices also need to be unblocked prior to reporting it missing, or it will deadlock in the mid layer.
- CQ 94747 - add changes required by kernel.org: (1) blk_req_bytes - there is a wrapper function for obtaining the data_len from the struct request object (2) DMA_BIT_MASK - newer kernels use this define for setting the DMA bitmask, it replaces DMA_32BIT_MASK and DMA_64BIT_MASK - without this change, the driver doesn't compile in 2.6.31 kernel.
- CQ 94401 - mpt2sas_transport_port_add: could not find parent sas_address: When direct attaching devices to controller populating every phy, the very last device fails being report with the error message "could not find parent sas_address". This failure is due to bug in _scsih_get_sas_address. This function was expecting the first phy of the controller to be handle zero, which is incorrect. Its actually handle 1. So the full range of handles assigned to the controller is 1 thru max phy count. So the sanity check in _scsih_get_sas_address was correct to include the max number of phys.

Major Changes For Version 03.255.02.00-1
Release Date: 08/10/2009

General Changes

- CQ 90217 - Time Stamp - needs to be set in IOC_INIT: Add support to set the TimeStamp when sending ioc_init. The driver will call do_gettimeofday kernel API to obtain the seconds+microseconds, which is provided to controller firmware.
- CQ 90438 - don't allow end user to set SATA queue depth above 32: Add sanity check in _scsih_change_queue_depth to limit the max_depth to 32 for SATA devices. This is only for physical devices not part of a volume.
- CQ 90485 - providing module parameter mpt2sas_fwfault_debug=1 results in panic: When there is BIOS, the driver would panic during driver load time when enabling the mpt2sas_fwfault_debug command line option. To fix this issue, the handling of mpt2sas_fwfault_debug is moved from mpt2sas_base_get_iocstate function to mpt2sas_base_hard_reset_handler. The issues occurs when the driver loads when the IOC is not in ready state. With a BIOS, the ioc loads in

operational state, hence the driver issues diag reset to put the controller back to ready state. By moving the code from one function to another, this new functionality is executed only when there is a diag reset issued from user space, or internal command timeout. This issue was introduced with CQ 37567, which is a new feature for enabling "stopping driver when Firmware encounters fault".

- CSET 92677 - Add Extended Type for Diagnostic Buffer Support: (1) Add extended buffer support in _ctl_diag_capability so this function returns success when ioc facts capabilities says firmware supports EXTENDED. (2) Add a print to send the message "Diag Extended Buffer" string to system log when ioc facts capabilities says firmware support EXTENDED. (3) Add support in the command line option diag_buffer_enable to support EXTENDED buffers when bit 2 is set. (4) Update comments section of the source code to indicate the buffer_type can be EXTENDED in addition to TRACE and SNAPSHOT.

Major Changes For Version 03.255.01.00-1
Release Date: 08/04/2009

General Changes

- CQ 64808 - FIXID support: new packaging component, containing KMP's, KMOD's, and DUD's.
- CQ 68144 - add option to automatically post DIAGNOSTIC trace and snapshot buffers when Linux driver loads: Added command line option diag_buffer_enable. When the command line option is set, the driver will automatically post DIAGNOSTIC buffers at driver load time. The command line option diag_buffer_enable is bitwise, so its possible to enable both and/or snapshot + trace buffers. For trace, the driver will allocate 1MB buffer, whereas for snapshot its 2MB. The purpose for this is so the end user doesn't have to manually use an application to setup DIAGNOSTIC buffers for debugging firmware related issues.

  Here is some examples

  trace:

  # insmod mpt2sas.ko diag_buffer_enable=1

  snapshot:

  # insmod mpt2sas.ko diag_buffer_enable=2

  both trace and snapshot:

  # insmod mpt2sas.ko diag_buffer_enable=3

- CQ 37567 - Support for stopping driver when Firmware encounters fault (1) Added command line option and shost sysfs attribute called mpt2sas_fwfault_debug. When end user writes a "1" to this parameter, this will enable support in the driver for debugging firmware timeout related issues. This handling was added in three areas (a) SCSI error handling callback called task_abort, (b) IOCTL interface, and (c) other timeouts that result in diag resets, such as manufacturing configuration pages. When this support is enabled, the driver will will provide dump_stack to console, halt controller firmware, and panic driver. The end user probably would want to setup serial console redirection so the dump stack can be seen.

  Here are the three methods for enable this support:

  (a) # insmod mpt2sas.ko mpt2sas_fwfault_debug=1
  (b) # echo 1 > /sys/module/mpt2sas/parameters/mpt2sas_fwfault_debug
  (c) # echo 1 > /sys/class/scsi_host/host#/mpt2sas_fwfault_debug (where # is the host number)

  (2) Fixed some of the comments sections for some of the function so "@ioc: pointer to SCSI command object" was changed to "@ioc: per adapter object"

- CQ 35970 - Create SYSFS SHOST attribute for sending task management request. (1) Create a sysfs attribute called task_management in /sys/class/scsi_host/host#. By witting a number to this attribute, it will do various types of task management. This code will send overlapping task management request to every device or pending task depending on the task type. This code was added to help debug firmware task management issues.

  Here is the supported numbers:

  1 = diag reset

  2 = message unit reset

  3 = abort task

  4 = target reset

  5 = lun reset
  6 = abort task set

example, to issue target reset to every device all at once:

echo 4 > /sys/class/scsi_host/host0/task_management

(2) Add more debug info for the MPI2_EVENT_SAS_DISCOVERY event when the MPT_DEBUG_EVENTS logging level is enabled. What was added was the string "start" and "stop", and the discovery_status info bits. (3) Add more debug when for the MPI2_EVENT_SAS_BROADCAST_PRIMITIVE event when the MPT_DEBUG_EVENT_WORK_TASK logging level is enabled. What was added was the string "primitive".

- CQ 14691 - Add support for RAID Action System Shutdown Initiated at OS Shutdown: (1) Add new function _scsih_ir_shutdown. This function will issue the MPI2_RAID_ACTION_SYSTEM_SHUTDOWN_INITIATED request via MPI2_FUNCTION_RAID_ACTION. The function will wait 10 seconds for reply message frame, then print out the ioc status and loginfo. This function is only called when there are raid volumes present. (2) Add shutdown callback in the struct pci_driver object scsih_driver. This will be called only when the system is shutting down. From this function, we will call _scsih_ir_shutdown mentioned above. (3) Add support in _scsih_remove to call _scsih_ir_shutdown. The function _scsih_remove will be called when the driver is unloaded (and system is still running).
- CQ 89677 - Linux SAS2.0 - Adding Phase 4.0 MPI Headers - revision L
- CQ 89640 - SAS2208/Thunderbolt support: Add support for PCI Device ID's range {0x80 - 0x87}
- CQ 86403 - running out of message frames: Due to the changes made in CQ 82588 to create a separate pools of memory for hi-priority and internal message frames, the driver doesn't have enough memory allocated for target mode task management request. To fix the CQ, the driver will allocate enough message frames to cover the number of command buffers in the internal message frame pool.
- CQ 85944 - Corruption caused by SAS Truncated CMD Frame: Add support in the driver to check for valid response info in the SCSI state, then check to see if the response code is MPI2_SCSITASKMGMT_RSP_INVALID_FRAME; when this condition occurs, the driver will return DID_SOFT_ERROR. A return code of DID_SOFT_ERROR will result in a retry at the SCSI-mid layer level. An additional change added to obtain the response code from the 1st byte of the response info instead of last.

Major Changes For Version 02.255.03.00-1
Release Date: 07/15/2009

General Changes

- CQ 82440 Drives not mapped on enclosure add after reset sequence The device driver was not handling updating device handles in all cases across diag resets. To fix this issue, the driver is converted to using sas address instead of handle as a lookup reference to the parent expander or sas_host. Also, for both expanders and sas host, the phy handle will be one unique handle. In the sas host case, the phy handle can be different for every phy, so the change is to set the handle to the handle of the first phy; every phy will be one single sas address(phy 0) instead of a different sas address for every phy(previous implementation). So making one consistent sas address for all the direct attached ports to the sas host, will make it better user experience when using udev /dev/disk/by-path dev nodes.
- CQ 80547 - IOCTLs fail after a SCSI passthru request times out The driver needs to call init_completion on a per request basis. At some point the wait_for_completion_timeout is not waiting for the timeout, instead returning immediately, thus going into diag reset. This fix will address all request using the wait_for_completion_timeout API. The previous implementation was only calling init_completion at driver load time.
- CQ 82718 - Inconsistent behavior when pulling/reinserting during IO (1) add support to spin up drives that are pulled and reinserted before device removal; the driver will delay putting devices in SDEV_RUNNING, following SDEV_BLOCK, until the drives have been spun up. (2) the device detection algorithm for "hot add" is enhanced; to include improvements surrounding devices that timeout on the first scsi commands attempts.
- CQ 79906 - 0402 Fault running heavy IO in I/T switching environment following a diag_reset, a request to send an ioc_init is timing out. The timeout occurred within the HANDSHAKE logic while waiting on firmware to acknowledge that the driver had wrote to the doorbell register. This was root caused to a logic timeout in the firmware code. The proposed solution is for the driver to call the udelay instead of msleep API in function where its looping reading the interrupt status. In addition to this change, there were two additional cases where we deleted the clearing interrupt status outside handshake context.
- CQ 84350 - Inaccessible drives not removed after zone configuration Add support to process device removal events when the phy status is set to MPI2_EVENT_SAS_TOPO_PHYSTATUS_VACANT.
- CQ 85702 - reported driver doesn't work in 2.6.24 linux kernel. The function pci_enable_device_mem was implemented in 2.6.25 kernel, so the drivers KERNEL_VERSION check needed to be changed from 2.6.21 to 2.6.25 for using the pci_request_selected_regions API.
- CQ 85974 - not all devices are being recognized while doing full cable pull testing This handles the case where driver receives an expander removal event while it is in the middle of processing an expander add event. The existing implementation will stop processing further device adds when a

expander delete arrives on top of an expander add. Due to a sanity check in the driver, the devices there were not added, were never notified to firmware with the device removal handshake protocol. Since the driver didn't do the handshake, the controller never provide further add events. To fix this issue, the sanity check was removed so the driver will always does the device removal handshake protocol.

Major Changes For Version 02.255.02.00-1
Release Date: 06/15/2009

General Changes

- CQ82588 - Add support for sending multiple task management request at the same time: (1) remove CHAIN_POOL support. The support is to put chains in a link list. This code was never used, so its being deleted. (2) create a pool of high priority message frames in the region of memory between message frames and chains. The modifications are in _base_allocate_memory_pools. Also create a separate pool of memory for internal commands located near the same region of memory. The pool of high priority message frames is restricted by the facts->HighPriorityCredit. (3) Create additional API for accessing request message frames. New function mpt2sas_base_get_smid_hpr is for highpriority request. New function mpt2sas_base_get_smid_scsiio for SCSI_IO, passing in the scsi command pointer. The mpt2sas_base_get_smid function is for requesting internal commands. (4) Added new function _base_get_cb_idx to obtain the callback index from one of the three pools of request message frames (mentioned in #3 above). (5) The function mpt2sas_base_free_smid was modified so the request message frames are put back on one of the three pools of request message frames (6) Added three new functions to handle sending target resest and OP_REMOVE from interrupt time, they are _scsih_tm_tr_send, _scsih_tm_tr_complete, and _scsih_sas_control_complete. This code will create a link list of pending target resets if there is no more available request in the hi-priority request queue. The list is stored in ioc->delayed_tr_list. (7) Added new function called _scsih_tm_tr_mp_send for mulitpath test case. This will send target resets over alternate path for DELAY_NOT_RESPONDING events. (8) Removed wrapper functions _scsih_scsi_lookup_set and _scsih_scsi_lookup_getclear. These were removed because this handling was moved into mpt2sas_base_get_smid_scsiio and mpt2sas_base_free_smid. (9) Add support to send task abort (instead of target reset) when there are timeouts from _scsi_send_scsi_io. This is less disruptive in multi-path environments. (10) Moved the allocation of ioc->pfacts up the code in mpt2sas_base_attach, because pfacts->MaxPostedCmdBuffers is required for target mode support when allocating memory pools.

- CQ 81462 - SLES11 not going into sleep state: (1) fix oops occurring at hibernation time. This oops was due to the firmware fault watchdog timer still running after we freed resources. To fix the issue we need to terminate the watchdog timer at hibernation time. (2) fix another oops occurring when the system resumes. This oops was due to driver setting the pci drvdata to NULL on the prior hibernation. Because it was set to NULL, upon resume we assume the pci drvdata is non-zero, and we oops. To fix the oops, we don't set pci drvdata to NULL at hibernation time. (3) fix problem with driver timing out on the first MPI request with interrupts are enabled. The problem is isolated to SLES10, only when MSIX interrupt routing is enabled. To fix the problem the driver will need to backup and restore the MSIX interrupt vector table going into and out of hibernation.
- CQ 81968 removing target mode source: deleted mpt2sas_stm.c and mpt2sas_stm.h. Removed target mode references in the driver Makefile.
- CQ 82466 - Adding DKMS KMP and external module support: (1) Adding mpt2sas-kmp.spec in the dkms kit, which is the spec file for KMP's. Adding dkms_post_add and dkms_pre_build scripts in the dkms kit. The dkms_post_add script will add mpt2sas string to ramdisk config files (etc/sysconfig/kernel for suse and /etc/modprobe.conf for Red Hat). The dkms_pre_build script will prep the linux kernel so the drivers will compile, required for compiling the drivers for the KMP rpms. (2) Adding Module.supported file in the duds, rpms and dkms so the driver is compiled with the external supported bit set in the modinfo (only for suse). With this fix, kits will not need to modify /etc/modprobe.d/unsupported-modules file. (3) Fix problem with Red Hat duds, where the driver was not provided on the dud, this was due to CQ 81968 (made in the previous release)

Major Changes For Version 02.255.01.00-1
Release Date: 05/21/2009

General Changes

- CQ 78919 Warm Plugging SAS behind enclosure doesn't bring drive back: The test case is drives are being powered on, and its taking roughly 15 seconds to spin up the hard drive. The device driver should be requeuing TUR's every one second to retry the TUR until the drive is no longer busy. However on SLES11, the queue_event API, is not waiting one second for the command to be retried. Instead, what we are seeing is the command is immediately sent, and there is no delay between retries, hence we cycle through the entire 144 retries in about 2-3 seconds. To fix the issue, we have to add a delayed_work member to the fw_event_work struct, and then call queue_delayed_work. In sles10/rhel5, we can call queue_work and queue_delayed_work using the same work_struct object. Since the 2.6.19 kernel the kernel maintainers separated the delayed work from

work_struct object, so you have to have separate work_struct and delayed_work objects.
- CQ 61087 - Target Mode commands lost with QD > 128: Numerous target mode enhancements were checked in, for instance unnecessary spin locks were removed, and kernel thread continually process new requests without going to sleep in between. Deleted single threaded kernel implementation. Fixed a mutex deadlock issue for non-data transfer request.
- CQ 78034 - update drivers to the MPI REV K headers: 1) remove VF_ID from all request descriptor API 2) rename VF_ID to msix_index in all the callback handlers 3) add VF_ID and VP_ID through out the code as "TODO" reminders in prior to mpi requests
- Update the copyright year to 2009 through out the code.

Major Changes For Version 01.255.03.00-1
Release Date: 05/15/2009

General Changes

- CQ 71559 - Target Driver is aborting an IO it shouldn't in I/T Switching Mode: The FW sends a reply frame for a task assist, then almost immediately reuses the same io index for another SSP_IU request. With the reply message frame just arriving for the previous TA, the driver kernel thread never went to sleep due to the new request. Apparently the kernel code didn't clean up the wait queue for previous command due to the nearly overlapping commands. So when the the new request arrives, there is no 5 second wait, and the driver is immediately aborting the command. To fix the issue, the driver will need to call init_completion() prior to every target assist.
- CQ 74873 – Target Driver Data corruption during heavy copy/compare stress test: The issue occurs when memory is being allocated for the lun ramdisk. There is a race between reads/writes to the same sector when dma_alloc_coherent called and when memory is available. To fix this, the driver will return SAM_STAT_BUSY prior to calling dma_alloc_coherent. From the initiator, when BUSY is returned, it will retry the command, by that time the memory will be available.
- CSET 75722 (CQ 75081): (1) add support to retry READ_CAPACITY16 when there is a unit attention (2) merge in changes requested by linux community with regards to T10 DIF, which is not to enable app tag checking for both type 1 and type3 protection.
- CQ 75352 - Power cycling cascaded expanders causes kernel panic: Code cleanup of the host reset handling of the driver, and fix hot plug work thread deadlocks, and kernel panics. (1) removed ioc->ioc_reset_in_progress flag, replaced with ioc->shost_recovery. (2) removed the spin lock around reading ioc->shost_recovery. (3) removed

changing shost state during host reset handling; such as SHOST_RECOVERY (4) moved rescanning the devices, updating handles, and deleting not responding devices to the same contents at the host reset handling. (5) return SCSI_MLQUEUE_DEVICE_BUSY from _qcmd when sas_target_priv_data->tm_busy is set (6) set SDEV_RUNNING from contents of the interrupt, instead of work threads (7) when there is a failure during _scsih_expander_add, need to call mpt2sas_transport_port_remove to properly tear down the port.

- CQ 76322 - RAID10 volume creation is showing as RAID 1E - Add support to obtain manufacturing page 10 at driver load time. The driver will display RAID10 string when the volume type is RAID1E, when there is a even drive count, and the manufacturing page 10 GenericFlags0 field has bit 2 is set.
- CQ 75917 When activating an inactive volume, the hidden raid components disappear. When a volume is activated, the driver will receive a pair of ir config change events to remove the foreign volume, then add the native. In the process of the removal event, the hidden raid component is removed from the parent. When the disks is added back, the adding of the port fails because there is no instance of the device in its parent. To fix this issue, the driver needs to call mpt2sas_transport_update_links() prior to calling _scsih_add_device. In addition, we added sanity checks on volume add and removal to ignore events for foreign volumes.
- CQ 78035 - TLR Support - query vpd page 0x90: The driver is sending vpd page 0, to get a list of supported pages. Then it will traverse the list searching for page 0x90. when page 0x90 is present, the driver will enable the TLR logic for Tape devices only. The TLR logic will enable the TLR bits in the SCSI_IO for every request. If there is a response with MPI2_SCSITASKMGMT_RSP_INVALID_FRAME, the driver will turn off the TLR logic.
- CQ 76771 - sas iounit config page0 timeout: 1) inhibit 0x3117 loginfos - during cable pull, there are too many printks going to the syslog, this is have impact on how fast the interrupt routine can handle keeping up with command completions; this was the root cause to the config pages timeouts 2) clean up interrupt routine to be more efficient. 3) clean up config page API - moving the setting and clearing of the mutex's to _config_request. There was a mutex deadlock when diag reset is called from inside _config_request, so diag reset was moved to outside the mutexs. Also deleted redundant code thru out the API, moving to common area; this reducing the code size by 1/3.
- CQ 73725 - ppc64 rpms not working: Added sanity check in spec file to check on the linux kernel. For ppc the name of the kernel is vmlinux-`uname -r`, whereas in other arch's its vmlinuz-`uname -r`. Also make sure the proper kernel naming is passed to mkinird.

- CQ 67203 - Gen2 rpm removal leaves mpt2sas directory behind. Add support to remove the entire mpt2sas directory instead of just the device driver KO file.
- CQ 77782 - dkms sles11 – unsupported-modules: Update driver spec file so for sles11 the allow_unsupported_modules flag is set to 1 in the file called /etc/modprobe.d/unsupported-modules. This will allow ramdisk creation.
- CQ 76728 - driver update disk doesn't work for sles11 and sles10: Root cause, the sas transport layer driver is not loading prior to our driver. To fix the issue, we add module.order and sas transport driver on the dud, and installer will know to load the sas transport driver prior to ours. Also on the sles10 dud, the UpdateId is changed so its unique identifier from the gen1 mptsas dud.

Major Changes For Version 01.255.02.00-1
Release Date: 04/13/2009

General Changes

- CQ 70172 - Adding EEDP support that uses the API in the Linux kernel (SLES11) (1) The driver registers for SHOST_DIF_TYPE1_PROTECTION and SHOST_DIF_TYPE3_PROTECTION. (2) For reads and writes needing DIF protection, the driver receives the SCSI_PROT_READ_STRIP and SCSI_PROT_WRITE_INSERT opcode in the scsi command. The driver will set the reference tag to the lower 32 bit of the lba, and application tag set to zero. The eedp flags are set to check/remove for reads, and insert for writes. (3) For MPI2_IOCSTATUS_EEDP_XXX errors, the driver will return check condition. The sense for GUARD_ERROR is 0x0B/0x10/0x01, for APP_TAG_ERROR is 0x0B/0x10/0x02, and for REF_TAG_ERROR is 0x0B/0x10/0x03.
- CQ 50118 - Add support for LUN RESET (SLES11). (1) Add new eh_target_reset_handler for target reset (2) Change the eh_device_reset_handler so its sending MPI2_SCSITASKMGMT_TASKTYPE_LOGICAL_UNIT_RESET, instead of MPI2_SCSITASKMGMT_TASKTYPE_TARGET_RESET. (3) Add new function _scsih_scsi_lookup_find_by_lun as a sanity check to insure I_T_L commands are completed upon completing lun reset.
- CQ 70577 - failed to obtain smid after 500 abort_task request The driver was not calling mpt2sas_base_free_smid() following a failed attempt in finding an active IO for an abort_task request via ioctls.
- CQ 67511 - query task for a driver selected task mid (1) rename function _ctl_do_task_abort to _ctl_set_task_mid (2) for both query and abort task,

call _ctl_set_task_mid(), which traverses the smid list looking for an active smid for the given device handle and lun.

- CQ 70399 - kernel panic on read of DVD with errors: Driver was corrupting memory by copying a large sense count into the SCSI command sense buffer following a MEDIUM_ERROR. The size reported in the reply mf was incorrect due to a bug in firmware. To prevent the kernel panic, the driver will calculate the sense buffer size by taking the minimum size reported in the reply mf, and SCSI_SENSE_BUFFERSIZE which is 90 bytes.

Major Changes For Version 01.255.01.00-1
Release Date: 04/02/2009

General Changes

- CQ 70194 - GEN2 - IOCTLS MAGIC NUMBER needs to be unique - Changed the magic number from 'm' to 'L' so RHEL4 work with register_ioctls32_conversion function.
- CQ 68327 - Append mpt2sas_ to the naming of all non-static functions - requested by kernel.org.
- CQ 70094 - removing the code associated to SAS_TRANSPORT_SUPPORT
- LINUX SAS2.0 -> KERNEL_ORG - little endian types going into mpi2_type.h Changed mpi2_type.h to use __le types, and fixed the rest of the driver where U64 was used, so it can directly access the u64 type, intead of double buffering into another __le64 type. There were some additional fix's in base_send_ioc_init, where u64 and u32 pointers are setup for the ioc_init struct, making use of BITS_PER_LONG to fix compiler warnings. Also merged in other coding style changes.
- CQ 69743 - Low Memory warnings and failed writes during target IO Removed a sanity check algorithm that would prevent allocating memory for ramdisk. Apparently the algorithm doesn't properly work on all systems when detecting the amount of available memory. On one system having 1GB of memory, the algorithm failed immediately. Since removing this code, the driver will allocate memory to the maximum memory available. However the fact of the matter is if you ask for more memory that is available, the linux kernel will send stack traces to logs, and there is no way to avoid this. Even those the logs are filled with stack traces, the system is still usable. Also, fix in the driver is to return MEDIUM_ERROR

for reads when we have ran out of memory. The prior code was returning ILLEGAL_COMMAND.
- CQ 68726 - GEN2 - target driver creates 4GB drives. The macro do_div was improperly providing the max_slices, which gave 4GB capacity for 32 bit OS, and 1MB for 64 bit OS. Changing the parameter from 32 to 64 bit solves the problem.
- CQ 70459 - handle release diag buffers when reset occurs when ioc operational. Add support to send MPI2_FUNCTION_DIAG_RELEASE when there is active diag buffers, and ioc is operational.

Major Changes For Version 00.255.13.00-1
Release Date: 03/25/2009

General Changes
- Adding SLES11(2.6.27.19-5) IA64 and Power PC support.
- CQ 69112 - Utilities like SLT MSM and LsiUtil hang when PD removed from VD. The driver was keeping around a sas_device object for a hidden raid component once it failed to be added because we try to detect the drive from the topo events. Then when the IR events arrive, the drive is not added because the old instance has stuck around. To fix the issue we don' t keep the old instance of the hidden raid component that was created during the topo events.

Major Changes For Version 00.255.13.00-1
Release Date: 03/18/2009

General Changes
- Adding SLES11(2.6.27.19-5) DKMS support. Modified driver compile script so drivers can compile from SLES11
- SCGCQ00067178 - (1) only request on msix interrupt handler(instead of 15) (2) when disabling interrupts, via System Interface Register, we need to do a readl following writel, because writes are posted. This will flush out the write.
- SCGCQ00068336 - import doorbell changes from the windows driver: (1) replace the MPI2_HIS_IOP_DOORBELL_XXX defines with MPI2_HIS_SYS2IOC_XXX and MPI2_HIS_IOC2SYS_XXX (2) towards

the end of the function, the driver needs to wait for the doorbell used bit to clear (3) the driver needs to clear the interrupt status register upon leaving this routine

- SCGCQ00068342 - Linux SAS2.0 - oops SMP_PASSTHRU when using MPI2_SMP_RT_REQ_PT_FLAG_IMMEDIATE: The bug was due to the driver assuming data was being passed thru the data_in_buf_ptr, which for immediate is NULL. For immediate the request data is appended to the message frame, in the place of the sgl.
- SCGCQ00068322 - fix compile error reported by Kernel.org: The compile failure was due to improperly defining _debug_dump_mf() function for when CONFIG_SCSI_MPT2SAS_LOGGING was not defined. Old: _debug_dump_mf(void *mpi_request, int sz); It should be: _debug_dump_mf(mpi_request, sz); Also fixed complier warning in mptctl.c, for task management completions.
- SCGCQ00068136 - SAS2 Linux driver does not release diag buffer on firmware fault. Following a FAULT, the IOCTLS which "releases buffers" fails. The reason its failing is the driver sends a request to FW to release the buffers, which fails, hence failing the IOCTL itself. The reason the mpi request fails is the FW has already released the buffers. The fix is to use a flag to indicate that its not required to send FW a mpi request to release the buffers.
- SCGCQ00066144 - Expander Reset causes drive-path to go away after several resets Fix race condition where the same target is being deleted and added on top of each other. Meaning the previous instance of the starget breakdown routine is called much later after the target was reported missing. Meanwhile the same target is being added. To fix this, we need to add additional checks in the target removal routine to insure the sas_device->starget is not being set to NULL when the same target is being added.
- SCGCQ00050744 - IO Pause running Target Reset TM during SMASH IO (1) Send VPD page 0x80, obtain the serial number, and save. Only done for disk peripherals (2) Search the sas topology for matching serial number. If found, then save links pointing to each other. (3) Delete links if one of the devices is deleted. (4) When there is a target_reset, then toss a ABRT_TASK_SET over to the other path(for every lun). This is done 5 seconds after the target reset. I found if you don't wait 5 seconds, I was still seeing command timeouts. It seems the disks are still initializing for some time after the target reset completion
- SCGCQ00068678 - SLIR2 - Driver Version not shown at correct place. Add "mpt2sas-" to the beginning of the driver_version string in MPT2IOCINFO IOCTL.
- SCGCQ00059724 - H200 program CR272561: modify mpt2sas to identify series-7 adapters at driver load time Modified _base_display_ioc_capabilities to print the Customer branding along with the VID, DID, SSVID, SSDID following the LSI branding that contains the card firmware/chip/bios versions. If the SSDID is not known but it is a

customer HBA, the driver will print the SSDID instead of the customer branding string. Nothing will be printed for other HBAs.

- SCGCQ00067144 - Driver crashed in mpt2sas:transport_port_add while hotplugging and removing disks The opps is in the sas transport layer, where they insure a phy that is being added to a port is not already binded to some other port. The fix is to save the port object inside each phy instance, then when a phy is being added, there is a new sanity check to make sure the phy is not previously binded. If its already binded, then release the phy binding before adding to a new port.
- SCGCQ00069472 - Linux SAS2.0 - Rapid Add/Delete - then ADD doesn't work. Plus Broadcast Primitive AEN fix. If a device is hot added, it normally takes several seconds for the drive to spin up; if the device is removed prior to the completion of this spinup time, the sas_device object will not be created and fail to be properly deleted when the remove event arrives. If the device is not send a target_reset, followed by a sas_iounit_ctnrl, upon device removal event, then the firmware will stop sending futher events for that device. So next time you add the device, there will not be an event. The fix is to add the sas_device object with only the handle and sas_address set. This will properly handle removing the device when the remove event arrives. Also there are fixes in the broadcast primative async event code where the driver would stop sending tm queries after the first query was completed. This was due to a bug the driver where it was not reseting the tm_cmds.status field back to MPT2_CMD_NOT_USED after completing a tm.
- SCGCQ00069351 - LUN Reset Task Management to target mode lun fails A bug in the code would return return a response code of TM_NOT_SUPPORT when there were no outstanding commands for task management request. The fix is to return TM_COMPLETE for the supported TMs, and still handle aborting the outstanding request.
- SCGCQ00069351 - Linux sas2.0 - make global symbols unique The driver was sharing the same ioc_list global parameter with GEN1 driver. The fix is to change all global parameters to have the mpt2sas_xxx prefix. Without this change, both gen1 and gen2 driver could possibly walk each other per host adapter struct.

Major Changes For Version 00.255.12.00-2
Release Date: 03/06/2009

General Changes

- Adding SLES11(2.6.27.19-5) standard RPM/DUD support for x86 and x86_64.

Major Changes For Version 00.255.12.00-1
Release Date: 02/27/2009

General Changes

- SCGCQ00062583 - Dud install fails in sles10. The driver was attempting to create the ramdisk in the incorrect folder. The fix is to add `cd /boot`in the update.post script prior to generating the ramdisk.
- SCGQ 20052, 24670, 63859 – various fixes for standard rpms. (1) The rpms installation folder was changed to copy drivers KOs to /lib/modules/`uname -r`/weak-updates, this was made to solve issues surrounding the backing up restoring mpt2sas drivers. (2) fix's bugs surrounding uninstalling the rpm (3) fixed copy error message when installing the rpm (4) fixed issue where this rpm overwrote the backup ramdisk image created by installing the gen1 mptsas driver. The solution is not to backup the initrd, but to recreate it when removing the rpm. (5) Added support to copy the driver sources tgz file to the /usr/src/redhat/SOURCE or /usr/src/package/SOURCE folders.
- SCGCQ00066888 – DKMS Packaging Requirements: (1) removed suse x86 prebuilt binaries (2) removed all the duds (4) removed RHEL5.0 and RHEL5.1 (5) removed SLES10.0 and SLES10.1 (6) upgraded dkms frame work to 2.0.21.1-1
- SCGCQ00064917 - task management fix's in ioctl path: (1) Set a mutex for task management request via IOCTLS (2) Quiesce IO when there was a phy hard and link reset via IOCTLS.
- SCGCQ00053879 - Linux SAS 2.0 - target mode driver oops when num_luns = 255: The oops was due to running out of memory during the creation of the ramdisk. Several fix's were made to address the memory footprint of the driver, for instance the driver will create 16k sgl element size, instead of 4k, hence reducing the size of the per command buffer scatter gather list size. The per command bufer ECHO_BYTES(4k) was moved from the per command buffer to the per lun struct. Semaphore added to insure atomic access to the ECHO_BUFFERs. Added a memory detection function to anticipate low memory, and to prevent asking for memory when its low, instead a check condition is passed back. Also, the ramdisk memory is allocated on the fly, instead of up front. There were several fix's made to address the driver failing on 32bit systems(SCGCQ00063721)
- SCGCQ00064886 - Gen2 - Target Mode (Loopback) functionality in Linux Moved the initialization of the target mode driver to before calling port_enable. Also removed a sanity check to prevent processing device add for sas_address that match's the controller.
- SCGCQ00064918 - Linux SAS2.0 wait for all pending commands to complete prior to diag reset Adding function call to _wait_for_commands_to_complete() which will wait as long as 3 seconds for all the pending commands to complete. This is called prior to issuing a reset.

- SCGCQ00063835 - Driver oops following firmware fault 7001: Fix driver oops that occurs during the timing out of port enable at driver load time. The oops is due to the driver delayed processing of events after the driver unloaded itself. The fix is to add a flag that will get set when port enable timeout. From event handling, a sanity check added for the setting of this flag.
- SCGCQ00050121 - LINUX SAS2.0 - End to End Data Protection: Remove the setting of the application tag.
- SCGCQ00064780 - Offline PD is coming as UG: The change is for the driver to ignore the MPI2_RAID_PD_STATE_OFFLINE event, insuring a bus/target is still assigned for the hidden raid component.
- SCGCQ00066234 - LINUX SAS2.0 - AENs for Linux and Windows behavior differences: Change code so driver always raises an AEN for MPI2_EVENT_LOG_ENTRY_ADDED; unmask event at driver load time. Also changed _ctl_release() so its never releasing the memory for ioc->event_log.

Major Changes For Version 00.255.11.00-1
Release Date: 02/09/2009

General Changes

- SCGCQ00063367 - adding mpi revision j support - (1) adding new headers (2) display MPI2_EVENT_SAS_DEV_STAT_RC_SATA_INIT_FAILURE as debug print (3) change naming of MPI2_IOCFACTS_CAPABILITY_PORT_ENABLE_EVENT_REPLAY to MPI2_IOCFACTS_CAPABILITY_EVENT_REPLAY

Major Changes For Version 00.255.10.00-1
Release Date: 02/04/2009

General Changes

- Adding RHEL5.3 Support.
- SCGCQ00036054 - Add Abort Task support for internal testing purposes - Add new function _ctl_do_task_abort(), which will search for an active smid for the given handle and lun number, then send off to firmware. If there is not an active smid, the driver will immediately return a mf reply filled out with termination count equal to 0.
- SCGCQ00061255 - Add MSIX Support - Add new functions base_enable_msix() and base_disable_msix() for enabling msix support, and failing back to io apic when msix is not support. Added function base_check_enable_msix() to detect msix support, and find out the max support vectors, and save base address to msix vector table. Add base_save_msix_table() and base_restore_msix_table() to address some

errata where the msix vector table was cleared following diag reset. Change command line naming for msi_enable to msix_enable. Fix oops occurring when unloading the driver due to events being queued when ioc->firmware_event_thread had already been destroyed.

- SCGCQ00050121 - Add EEDP(End to End Data Protection) Support - (1) detect EEDP presence, (2) media formated for EEDP commands. This is done from scsih_slave_configure(), we check the PROTECT bit in byte 5 of the inquiry data, then sending READ_CAPACITY_16 which contains the protect type and block size. This information is saved in the per LUN private data. When SCSI command are sent from scsih_qcmd, the EEDP presence is checked, then the SCSI opcode is checked against an array of supported EEDP commands. After passing a series of sanity checks, the SCSI_IO request is set for EEDP transfer; filling in the reference tag with the lower 32 bits of the LBA, and application tag with the lower 16 bits of the LBA. In the completion routine scsih_io_done we check for IOC_STATUS EEDP_GUARD_ERROR, EEDP_REF_TAG_ERROR, and EEDP_APP_TAG_ERROR, returning parity error(DID_PARITY). A command line option cm_target was added so crack monkey EEDP target can be used for test purposes (which bypasses some of the sanity checks).
- SCGCQ00061428 - Fix compile error in target mode source: Delete the extra "{" in the function __stm_do_mod. This function is only compiled into i386 builds.
- SCGCQ00062709 - misc bug fixes - (1) From config_get_sas_iounit_pg1(), the config page number was getting set to zero, instead of one, hence the incorrect config page data was being returned. (2) Set the device state to SDEV_RUNNING when previously SDEV_BLOCK following host reset. With out this fix, the devices were staying in SDEV_BLOCK state. (3) For SCSI_IO with a return of IOCSTATUS_IOC_TERMINATED, the driver needs to check if the device state is SDEV_BLOCK state, then always return DID_BUS_BUSY. Previously it would return DID_RESET, or return DID_BUS_BUSY for loginfos 3117000. This change will be uniform the expected return when the path was lost. For some multi-path solutions that expect DID_BUS_BUSY when the path is lost.
- SCGCQ00061900 - Linux driver will not function in simultaneous Initiator and Target Mode - (1) The first target assist was failing with MPI2_IOCSTATUS_TARGET_DATA_OFFSET_ERROR due to some stale data in the Relative Offset. The bug was due to memset() the request message frame with the incorrect struct size. (2) The driver was running out of message frames due to it not allocating extra request frames for the target command buffers(128). The fix was to report less available request message frames to the scsi midlayer for sending initiator requests. The change was to set the shost->can_queue value to 362 from the previous value of 490. This is only valid with the driver is compiled to

support target mode. With out target mode support compile in, the shost->can_queue will be set to 490.

- SCGCQ00060619 - When issuing PhyResets and disabling and enabling phys to Bobcat The test case was hitting an oops due to accessing an NULL pointer. This bug was in _scsih_mark_responding_sas_device, where it was calling starget_printk without making sure starget was non-zero. The same fix was made in _scsih_mark_responding_raid_device. The effected code is used to check whether device handles have changed following diag reset.
- SCGCQ00062743 - Merge diag reset code in from the windows driver (1) Change the wait time to 50 ms from 100 ms following writing MPI2_DIAG_RESET_ADAPTER to the Host Diagnostic register. (2) Rewrite the 300 second wait loop following writing MPI2_DIAG_RESET_ADAPTER to the Host Diagnostic register (3) Change the wait time to 20 seconds from 60 seconds when waiting for the READY ioc state following reset.

Major Changes For Version 00.255.09.00-1
Release Date: 01/19/2008

General Changes

- SCGC000057110 - The module load crashed when module is reloaded after previous error load . The fix is to call scsi_remove_host from scsih_probe when base_attach returns failure. Also we need to delete the instance of ioc from ioc->list link list prior to calling scsi_remove_host. The reason is the memory allocated for ioc object is freed from inside scsi_remove_host.
- SCGC000053522 - Diag reset under heavy IO load causes massive data corruption (reopen). After diag reset, its possible that device handles are getting changed. The previous fix in 0.255.08.00 was only updating the handles in struct _sas_device. This was not the complete fix. In addition, the driver needs to update the handles in the starget host data in struct MPT2SAS_TARGET.
- SCGC00059765 - SLES10 KERNEL PANIC - The oops was due to general protection fault when accessing an invalid address reply message frame. To fix the problem, the driver needs to poison (0xFFFFFFFF) both upper and lower 32bit words of the reply descriptor when completing the request, then sanitize upon interrupt to insure they are not all FFs. The problem is there is a race between the driver reading the reply post descriptor pool, and data being flushed across the PCI bus to host memory. After an analysis of PCI traffic, it seemed the driver was reading the descriptor while it was in the middle of being being updated by firmware. Meaning the lower 32bit had been properly updated, whereas the upper 32 bit word was its previous value from the last write.

Major Changes For Version 00.255.08.00-1
Release Date: 12/31/2008

General Changes

- SCGC000053173 - SAS 2.0 Linux - checkpatch.pl fix's.
- SCGC000053130 - Kernel panic on enclosure hot plug. Removed some code that was added for debugging CQ 41541. The code was looking for matching enclosure logical id and slot when a device was added. When there was a match, the kernel would panic. This code will hit the panic when there is multipath, so it needs to be removed.
- SCGC000053149 - SAS 2.0 Linux - target mode fix's. (1) fix big endian bug in stmapp_set_response_info() - setting the proper response code. (2) set the proper sense data response length in stm_send_target_status()
- SSCGC000053522 - Diag reset under heavy IO load causes massive data corruption. The problem was device handles were changing after diag reset, resulting in IO going to the incorrect device. The issue was fixed by holding off IO untill after the device handles had been refreshed. Also put the controller shost_state into SHOST_RECOVERY during diag reset, which will better emulate the behavior of scsi mid layer invoked host reset. Add sanity checks following reporting device to scsi midlayer. This is to insure the discovery of the device occurred successfully. When not successful, the driver will remove the instance of the device from the internal link lists(either sas_device or raid_device).

Major Changes For Version 00.255.07.00-1
Release Date: 12/11/2008

General Changes

- SCGCQ00051110 - Unable to install RH5.2 with Gen2 BIOS and 255.06 driver: The first INQUIRY sent MAXTOR drives is failing with ILLEGAL_REQUEST, and the device doesn't get discovered. The reason for this failure is untagged commands were being sent which are are not supported by these particular drives. This bug was introduced by SCGCQ00046290, where a shared flags variable was getting cleared when the MPT_DEVICE_TLR_ON was suppose to be getting set. This caused the MPT_DEVICE_FLAGS_INIT bit to get cleared, hence untagged was sent instead of simple. The fix was a matter of ORing instead of ANDing.
- SCGCQ000502063 - upon volume creation reboot is necessary to access drive: Add support for new events recently added in firmware: RC_VOLUME_CREATED, RC_VOLUME_DELETED,

RC_PD_CREATED, and RC_PD_DELETED. Add support for MPI2_EVENT_IR_VOLUME and MPI2_EVENT_IR_PHYSICAL_DISK, which provide status change info, needed for handling volumes and pds that transitioning between online and offline/missing states.

- Added extra print in _ctl_do_mpt_command to display the request mf when there is a timeout.
- SCGCQ00047075 - Deletion of Hot Spare drive does not unhide the drive for linux. This issue fixed indirectly by CQ 502063.
- SCGCQ00046684 - MPT2BTDHMAPPING IOCTL does not return valid devHandle for IR Volumes: Add support for searching the raid_device_list. Fixed bug where we failed to find corresponding target/id when passed a valid handle. This is due to incorrect pattern match on target/id. We where expecting 0xFFFF, when it should of been 0xFFFFFFFF. This result of this bug is lsiutil application fails to display hidden raid components from option #8.
- SCGCQ00045524 - drive swap during reset causes stack dump: The oops says the previously reported phy is being reported again. The bug is due to stale data being kept in the internal driver sas topology tree. When the drive is being added to a different slot after being removed, the driver is reporting the old instance of the phy, in addition to the new phy. So when the 2nd drive is added, the phy instance has been already added, resulting in the oops. To fix the issue, we clear the remote_identify data from the phy instance when the drive is removed. The fix is located in transport_port_remove.
- Add _debug_dump_mf function in mpt2sas_debug.h. This is called to dump the message frame contents when there is a timeout.
- Separate _scsi_remove_unresponsive_devices, into three separate functions. From this new calls, we only scan the config pages once to determine which devices are still present. The old implementation would loop on rescan with every device, adding to redundant config page request.
- Add spin locks in _scsih_sas_device_remove.
- For RAID volumes, added sanity check after reporting the volume to make sure volume was successful in being added. If it failed to get added, we removed the instance of the volume from the raid_device_list.
- Added BUG() in _scsih_add_device. This will oops the driver when sanity check matched an enclosure logical id and slot in sas_topology. This to insure we are not adding a device that was previously added. This should never be hit.
- SCGCQ00041541 - Snowmass: Segmentation Fault loading the driver: The oops says the previously reported phy is being reported again. The bug is due to the bobcat SEP device being reported twice. This is a direct effect of timeouts occurring to unsupport multipath target mode device, which result in a diag reset. The diag reset replays the sas topology events, so this event is processed, as well as the "start of day" handling of adding the device. So to fix this issue, we are adding a sanity check

topology events handling to see whether the "start of day" handling is pending, if so then ignore.

- Fix a oops when unloading the driver when there is two host adapters. The oops was replicated when one of the controllers faulted loading. When this occurs, the sas_port_list for the sas_hba was not initialized, then when the driver unloaded we hit the oops. To fix this issue, the initializing of the sas_port_list needed to be moved to scsih_probe.
- SCGCQ00053026 - LINUX SAS2.0 - Add support for MPI2_IOCFACTS_CAPABILITY_PORT_ENABLE_EVENT_REPLAY: The driver will issue MUR when it loads, unloads, and from the power management callbacks. All other places the driver will issue diag reset for recovery. The driver will check the capability prior to sending MUR.

Major Changes For Version 00.255.06.00-1
Release Date: 11/21/2008

General Changes

- SCGCQ00046278 - Linux SAS 2.0 - dma_get_required_mask doesn't work correctly. Due to a bug in dma_get_required_mask, its possible that linux kernel would report 32 bit physical addressing, when in fact there was over 4gb of available memory, thus resulting in a performance hit. A work around for this problem is under 64bit systems, we will need to set the requested dma mask prior to calling dma_get_required_mask. This issue was eventually fixed in newer 2.6.27 kernels, and this workaround not needed.
- CSET SCGCQ00046290 - Linux SAS2.0 - SAS-2 Host TLR Requirement Enhancement - Enable/Disable TLR (1) removed sending scsi opcode 0xC2 for HP tapes at driver load time (2) added MPT_DEVICE_TLR_ON bitwise define. This is set in the sdev_private_hostdata. This bit is set for all SSP devices at driver load time from slave_alloc(). (3) added tlr_snoop_check. This is flag to indicate the first command issed to a device. This is used when snooping the first command response. If the response code equals MPI2_SCSI_TASKMGMT_RSP_INVALID_FRAME, then turn off MPT_DEVICE_TLR_ON bit. (4) for all SCSI_IO request, check for MPT_DEVICE_TLR_ON bit, then set the MPI2_SCSIO_CONTROL_TLR_ON bit in the SCSI control field.
- SCGCQ00037569 - Linux SAS2.0 - Target Mode drivers - need to remove support for NextCmdBufferOffset in CommandBufferBaseRequest moved all the private driver data in the per command buffer request area to its own structure `struct CMD` made the command buffer area 64 bytes.
- SCGCQ00046560 - Linux SAS2.0 - Target Mode compile fix's for 32bit OS added support for divide 64bit words in 32bit OS added support for modo 64bit words in 32bit OS

- SCGCQ00041760 - Target system completely locks up when loading Target Mode driver after Initiator loaded Fix oops: The target mode driver is receiving a MPI2_EVENT_HARD_RESET_RECEIVED event during the 1st port enable, which results in NULL pointer dereference. The pointer "priv" is initialized from stm_adapter_install, which is called after port enable. A sanity check has been added prior to accessing the "priv" pointer to insure its non-zero.
- SCGCQ00045607 - incorrect I/Os aborted: The fix is when the task_management request arrives, the driver will immediately mark the IO as IO_STATE_ABORTED, then it abort any active task assist. The commands that have not been sent a target assist, will be intercepted when its thread is activated, from a sanity checks on the IO_STATE_ABORTED flag (those IOs will be aborted). Finally, when the task_management thread it activated, the remaining IOs will be aborted while matching IO_STATE_ABORTED bits and tm task tags.
- SCGCQ00038858 - Initiator/target returns constant Queue Full response - The problem is firmware is sending new command buffers while the previous command buffer work thread is still active. To fix this issue, the driver will be checking the IO_STATE_POSTED bits at the end of the work thread to see if another command buffer has arrived, then loop back to work the next.
- SCGCQ00037650 - Kernel panic on phy pulls with target mode IO - This problem occur as a result of commands timing out, and task management threads being activated. The oops show the problem in base_get_smid. Apparently the driver was calling base_free_smid() twice for the same smid, thus corrupting the link list pointers. To fix this issue we removed the redundant calling of base_free_smid.
- SCGCQ00050114 - SATA QUEUE DEPTH - (1) Change SATA Queue Depth to 32 (2) Change RAID10 Queue Depth to 128.

Major Changes For Version 00.255.05.00-1
Release Date: 10/20/2008

General Changes

- SCGCQ00042225 - Linux SAS2.0 - Add support for MPI Headers (v2.00.10) - Revision I: (1) MinReplyFrameSize was changed to ReplyFrameSize in the ioc facts. (2) SystemReplyFrameSize was removed in ioc_init. The driver will set this in only older firmware. (Both reply and request frame size are the same). (3) MPI2_SCSIIO_CONTROL_UNTAGGED was deleted. (4) NextCmdBufferOffset was removed in CmdBufferPostBaseRequest. (5) StatusDataSGL was renamed to StatusDataSGE in TargetStatusSendRequest.

- SCGCQ00040580 - Linux SAS2.0 - adding SMP Portal support for SLES11/RHEL6 kernels: added transport_smp_handler() which is a transport portal for smp passthru, it can be used by using smp_utils example: smp_rep_general /sys/class/bsg/expander-5:0
- SCGCQ00037564 - Linux SAS2.0 - Diag Buffer Support
- SCGCQ00042598 - Unable to use 32-bit LSIUtil with 0.255.04 driver - Fix bug in copying 32 bit pointers to 64 pointers in the compatibility section of the IOCTLS: Pointers (reply_frame_buf_ptr, data_in_buf_ptr, data_out_buf_ptr, sense_data_ptr).
- SCGCQ00043467 - Bump Driver Version

Major Changes For Version 00.255.04.00-1
Release Date: 10/10/2008

General Changes

- SCGCQ00040688 - Continuous reset with Snowmass topology: In IOCTLS, add sanity check to insure the device handle is non-zero.
- SCGCQ00040725 - on cable pull/push, synchronizing cache for a missing drive results in diag reset: (1) Add new functions which will set the sdev state to SDEV_BLOCK when the firmware sends DELAY_NOT_RESPONDING events, and set sdev state back to SDEV_RUNNING when the device is added within device loss timeout. This will help with handling fail over fail back in multipathing environment. (2) Add new functions which will per device flags when there are active task management requests, this flag will inhibit IO only to that device. Previous implementation will inhibit IO to every device when there was task management(even if it wasn't involved). This effected performance in multipathing environment during fail over. (3) Fix bug where a device is not completely removed from internal driver topology structures when file handles are open. This will result in not allowing a device to be hot added after it was previously mounted. (4) Fix bug when direct attached devices are not detected during device add.

Major Changes For Version 00.255.03.00-1
Release Date: 10/07/2008

General Changes

- Change driver versioning to SAS2.0 firmware protocol.
- SCGCQ00037764 - Linux SAS2.0 - QUEUE_FULL - raid support : We need to send scsi_track_queue_full() to the volume when QUEUE_FULL event arrives for the hidden raid component. This will throttle back the number of outstanding commands sent to the volume.
- SCGCQ00035929 - Linux SAS2.0: TLR enabled in Gen 2 Drivers: Enable TLR bits in SCSI_IO control field for TAPE.

- SCGCQ00020011 - Linux Driver - Gen 2 - Add Poll Support: (1) Add _ctl_poll(filep,wait) function, which is the callback handler for polling. (2) From ctl_release(inode, filep), added code to free the memory assoicated to event log when all applications are unloaded. (3) From ctl_add_to_event_log(), removed the sanity check for asyn_queue, moving down to the SIGIO call. (4) From ctl_add_to_event_log(), add wake_up_interrruptible. (5) From ctl_init, initialize the waitqueue "ctl_poll_wait". (6) From ctl_eventenable, clear aen_event_read_flag, and delet the same code from ctl_fasync.

Major Changes For Version 00.00.02.19-1
Release Date: 09/19/2008

General Changes

- SCGCQ00037575 - (1) Big Endian fixs surrounding writeq. (2) Big Endian fixs in target mode function stm_cmd_buf_post_base. (3) PPC compile warning fix's surrounding sas_address. (4) Cable Breaker enhancements to debounce multiple expander add and delete events. (5) Cable Breaker enhancements during link status change events so sas_device_pg0 is not read when the link is zero. (6) fix bug where internal scsi_cmds.status flag bit "MPT2_CMD_PENDING" was not cleared in error situation. (7) fix bug where we were not allocating enough memory to contain all the chain buffers. (9) delete ioc->reply_depth, replacing with ioc->reply_free_queue_depth.
- SCGCQ00036954 - Linux SAS2.0: BTDHMAPPING IOCTL , TargetID and BUS changed to 32 bit
- SCGC000036240 - Linux SAS2.0 - incorrect request frame size: Driver sending the incorect reply/request frame size duing ioc_init. The reply and request frame sizes were set to the other.
- SCGC000036256 - SAS2.0 - incorrect calculation of sg_tablesize when the max chain depth is less than 14: the sg_tablesize calculation for when MaxChainDepth was less than 14 was using initialized data.

Major Changes For Version 00.00.02.18-1
Release Date: 09/11/2008

General Changes

- SCGC000033874- Devices in cascaded JBODs are not detected: from _scsih_expander_add(), add sanity check to insure parent has been already added.
- SCGC000033457 - Kernel panic on enclosure power off after errors: from scsih_qcmd(), add sanity check on scmd->device->hostdata NULL pointer.

- SCGCQ00034008 - constant mpt2sas log_info: Added "ioc-> tm_cmds_busy" flag which is set in all branches of the code when task management is being active, then Inhibit the following loginfos : {0x30050000, 0x31140000, 0x31130000}.
- SCGCQ00031288 - Constant device/expander config page 0 read: from scsih_add_device(), prevent initiator from being added as a device; from _scsih_queue_rescan(), don't allow rescan if diagnostic reset occurs during initial port enable; from _scsi_remove_unresponsive_devices(), fix a bug which resulted in infinite loop execution.

Major Changes For Version 00.00.02.17-1
Release Date: 09/02/2008

General Changes

- SCGCQ00033819 - Adding support for 2.6.25 and 2.6.26 kernels (a) scsi_command changes - single request_buffer was removed, and replaced with common API, for (scsi_sglist, scsi_dma_map, scsi_dma_unmap), Also added support for new scsi accessors. (b) support for pci_request_regions (c) shost attributes changed from class_device_attr to device_attr (d) merged in work thread API changes, and removal of KOBJ_NAME_LEN
- SCGCQ00033820 Fix OOPS - two liberators connected. One running in target, the other initiator. The oops was due to the windows initiator having target mode support enabled. With it enabled, the Linux target tried to add the target. Since the target failed to be detected, the driver deleted the device in the device link list, however the port was still present in the topology. When the windows initiator rebooted, the driver tried to add the same port again, and that caused an oops in the Linux kernel. To this this we have to fully break down the port when the device was not detected.
- SCGCQ00034011 SAS2.0 - target mode - fix OOPS when using command line option num_luns > 2. The driver was only allocating up to NUM_LUNS pointers for the disk array. Since NUM_LUNS is set to 2, anything larger would be corrupting data. The fix is to increase NUM_LUNS to 511, and set the default for the command line option to 2. If the command line option was larger than 511, then it would set the max to 511.
- SCGCQ00034018 - SAS 2.0 - target mode - dev testing diag reset fix's (a) need to repost command buffers following host reset, (b) need to initialize all the command buffers status; e.g. "ioc_cmd_priv->status" to MPT2_CMD_NOT_USED (c) when tearing down the kernel threads at driver unload time, need to set flag "priv->stopping_threads" flag. This flag will prevent driver from sending stale request.

Major Changes For Version 00.00.02.14-1
Release Date: 08/27/2008

General Changes

- Adding Target Mode Support. This support is disabled by default. You will have to recompile the driver; enabling support by modifying the driver Makefile, uncommenting this line:
  "EXTRA_CFLAGS += -DTARGET_MODE".
  There new command line options for tuning the behavior of the target mode driver:
  1) num_luns – sets the number of luns, by default its 2.
  2) disk_chunks – sets capacity of each lun, by default 50MB.
  A new logging level was added for which enables debugging of the driver. Its is MPT_DEBUG_TARGET_MODE found in mpt2sas_debug.h.

Major Changes For Version 00.00.02.08-2
Release Date: 07/21/2008

General Changes

- Defect – SCGCQ00024670 – rpm install/update posts cp error messages on RHEL5 U2 SAS2 driver. A sanity check was added to insure the file was there before we copy it.

Major Changes For Version 00.00.02.08-1
Release Date: 07/11/2008

General Changes

- Enhancement – SCGCQ00020756 – Initial target mode framework.
- SCGCQ00020415 - mpi2: 02.00.09 header files
- SCGCQ00016671 - Reporting devices listed in BIOS PG2 ahead of all other devices
- SCGCQ00018908 - RPM Install Failure
- Adding RAID recognization and hotplug support.
- Bug fix in fault polling handler where most significant bit of the doorbell register was being removed, thus preventing recovery.
- Rename __FUNCTION__ to __func__.

Major Changes For Version 00.00.02.04-1
Release Date: 06/23/2008

General Changes

- Enhancement – SCGCQ00017004 - Create separate logging level for firmware event work task

- Enhancement – SCGCQ00018680 – Added displaying verbose info for when firmware returns non-zero iocinfo
- Enhancement SCGCQ00019136 – 1) Added tune-ables for setting controller queue depth and scatter gather element depth per IO, 2) Added check for pci_dma_mapping_error when user buffers mapped to mapped to physical memory, 3) Moved obtaining static configuration pages to prior to port enable, 4) Added command line option to disable MSI, 5) Added device sysfs attributes : device handle and sas address, 6) Added host sysfs attribute: sas address
- Defect SCGCQ00019066 – 1) SimpleTags are used by default for commands sent prior to slave_config, 2) Many bug fixes in function _scsih_wait_for_device_to_become_ready, 3) Added _scsih_start_unit, this to be called from _scsih_wait_for_device_to_become_ready, when test_unit_ready returns sense data [6/4/11], 4) Added scsih_display_sata_capabilities, which displays SATA features from slave_config, 5) Moved reporting devices during the initial port enable to the end of mpt2sas_probe, and use sas_device_init_list which is a separate link list only used during init, also added the flag wait_for_port_enable_to_complete, to handle this branch, 6) In struct fw_event_work, add a "retries" field for each device to the retries can be tracked separately for each individual device, 7) scmd->resid - was not being set in the scsih_io_done function, 8) For a REPORT_LUNS completion when there is no data transfer, change the return code to sense ILLEGAL_COMMAND, thus avoiding an ugly log to var/log/messages 9) From _scsih_sas_topology_change_event, removed deleting devices and expanders when firmware returns XXX_DELAY_NOT_RESPONDING
- Enhancement SCGCQ00019755 – 1) Upgraded dkms framework to 2.0.19.1-1, 2) Removed 32bit SuSE support from the dkms rpm.

Major Changes For Version 00.00.02.00-1
Release Date: 05/23/2008

General Changes

- Add RHEL5.2 and SLES10 SP2 support
- Add MPI2.0 Rev H headers
- Enhancement – SCGCQ00014681 - LINUX: smart predicted fault
- Enhancement – SCGCQ00014677 - LINUX: support for atomic 64 bit write to MMIO – A new function called base_writeq was added. This is the glue for handling atomic 64bit write to MMIO. This special handling takes care of 32bit environment where its not guaranteed to send the entire work in one transfer.

- Enhancement – SCGCQ00014688 - LINUX: enable firmware queue full handling – driver will either enable/disable firmware task_set_full handling in iounit_page_1. Added support for MPI2_EVENT_TASK_SET_FULL. Set SAS queue depth to 264, and SATA/STP to 8. Added prints to display whether NCQ is enabled.
- Defect – SCG00016706 - Fix panic when unloading driver with no attached devices – the panic occurs in scsih_remove because sas_host object has not been initialized, due to code relying on sas_topology events. When there are no attached devices, the firmware doesn't send sas_topology events. The fix is to create the sas_host object upon the first discovery start event.
- Defect – SCG00016713 - IOCSTATUS_INVALID_SGL - when writing to iounit page 1 – The fix is to set MPI2_SGE_FLAGS_HOST_TO_IOC bit in the sgl.
- Defect – SCG00016736 - LINUX: fix compile issue in RHEL5.2 kernels – the fix is to rename shost_priv to shost_private.
- Defect – SCG00016740 - LINUX: fix compile errors when adding new MPI2.0 Rev H, 2.00.08 headers – the fix is to rename Mpi2SSepRequest_t to Mpi2SepRequet_t.

Major Changes For Version 00.00.01.00-1
Release Date: 05/15/2008

General Changes

- Enhancement - SCGCQ00014683 - LINUX: interrupt mask register – There are two existing API in the driver for masking and unmasking interrupts. They are called _base_mask_interrupts and base_unmask_interrupts. The change is we will first read the HIM register, then logical OR the bits with the mask that we want to change, then write it back to the HIM register.
- Initial release.