SAS2.0 Linux OS Driver Release Notes

Driver Version: mpt2sasbtm-14.00.00.00-1

06/29/2012

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.

Special Notes For This Build:

- This Build supports only RHEL4.6, RHEL4.7, RHEL4.8 and SLES9 SP4 Operating Systems only.
- This build includes binaries for X86, X86_64 and PPC64 architectures only.
- The diskdump will not work if the controller is in a non-operational state.
- There is a problem in the fixid kit based DUD which prevents using it for DUD based installation.

Current Errata

- None

Driver Release Package Contents

• *Source tarball*

Source tarball is named as: mpt2sasbtm-<version>-src.tar.gz

where:
<version> = version tag for this rpm

• *RPM Binaries:*

RPM images are named as: mpt2sasbtm-<version>-<release>-<os>.<arch>.rpm

where:
<version> = version tag for this rpm
<release> = release tag for this rpm
<os> = {rhel4}
<arch> = x86_64 - Opteron Processor, w/ x86_64 install
i686 - i686 or later processor (Red Hat)

• *Driver Update Disks*

dud images are named : mpt2sasbtm-<version>-<release>-<os>.<arch>.dd.gz

where:
<version> = version tag for this rpm
<release> = release tag for this rpm
<os> = {rhel4}

<arch> = x86_64 - Opteron Processor, w/ x86_64 install
i686 - i686 or later processor (Red Hat)

- *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 RHEL4.7 release, 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:

I586
- SLES9
  - SP4 (2.6.5-7.308) (bigsmp, debug, default, smp)

x86_64
- SLES9
  - SP4 (2.6.5-7.308) (default, smp)

ppc
- SLES9
  - SP4 (2.6.5-7.308) (pseries64)

ia64
- Not supported in this release

Red Hat:

i686
- RHEL4
  - Update 6 (2.6.9-67.EL) (normal, hugemem, smp)
  - Update 7 (2.6.9-78.EL) (normal, hugemem, smp)
  - Update 8 (2.6.9-89.EL) (normal, hugemem, smp)

x86_64
- RHEL4
  - Update 6 (2.6.9-67.EL) (normal, largesmp, smp)
  - Update 7 (2.6.9-78.EL) (normal, largesmp, smp)
  - Update 8 (2.6.9-89.EL) (normal, largesmp, smp)

ppc
- RHEL4
  - Update 6 (2.6.9-67.EL) (normal, largesmp)
  - Update 7 (2.6.9-78.EL) (normal, largesmp)
  - Update 8 (2.6.9-89.EL) (normal, largesmp)

ia64
- Not supported in this release

Major Changes For Version 14.00.00.00-1
Release Date: 06/29/2012

General Changes

- CQ 300005: Bump driver version for phase 14 GCA release

Major Changes For Version 13.254.02.00-1
Release Date: 05/30/2012

General Changes

- CQ 289489: Bump driver version for phase 14 beta release

Defect fixes

- CSET CQ282260: Providing a negative value to sdev_queue_depth parameter leads to incorrect queue depth for a SAS drive. When the driver is loaded by providing a negative value for the SAS queue depth (sdev_queue_depth) parameter, then the qdepth value for the SAS devices is incorrect. Fix is to cast the unsigned integer variable sdev_queue_depth to an integer before comparing it with a negative value.
- CSET CQ 286976: If the requested IOCTL block state is equal to NON_BLOCKING and mutex_trylock is succeed, then control flow goes to mutex_lock_interruptible() which will deadlock. To fix this, code in _ctl_ioctl_main is rearranged so the success of mutex_trylock is not calling mutex_lock_interruptible().
- CSET CQ 286984: Infinite TUR command retries happen when a bad drive is connected. When a drive that continuously sends Check Condition status, with SenseKey/ASC/ASCQ as 0x02/0x04/0x00 (indicating that the device is not ready) is connected, then it is observed that the driver continously sends TUR commands to the drive, without ever signalling an error. The return code of the function which issues the TUR command was not being checked

properly for number of retries. Modified this code to correctly check the return code and indicate an error if the number of retries is exhausted.

Major Changes For Version 13.254.01.00-1
Release Date: 04/02/2012

General Changes

- CQ255068: The customer branding string for the "SSD 910 Series" controller was udpated.
- CQ255069: The copyright string in all the driver sources was changed to 2012.
- CQ264671: Bump driver version to 13.254.01.00 for phase 14 pre alpha release
- CQ264678: MPI 2.0 Release: MPI 2.0 Rev V (2.0.14) specification and 2.00.25 header files

Defect fixes

- CQ252479: Target IDs of the removed drives are taken by the existing drives/enclosure in the topology. Following host reset, firmware discovery is reassigning another hard drive in the topology to the same device handle as the device that is getting hot removed. When the driver device removal routine is called, there will be two hard drives with the matching device handle in the internal device link list. In the device removal routine, we call a separate function which is removing the device. Since this routine is having the device handle passed in as input parameter, the routine will be traversing the internal device link list searching for matching device hande. Since there will be two devices with matching device handle, an incorrect device could be removed. To fix this issue, the device is removed in the device removal routine, instead of calling the function that removes the drive based on the handle.

- CSET CQ259044: Driver oops associated with the max_queue_depth command line option set too small. The driver change is to set the scsi mid layer can_queue value equal to the max_queue_depth. Then the overall message frames required IO is the minimum of either (max_queue_depth plus internal commands) or the IOC global credits.
- CSET CQ259281: The log_info constant is supposed to be IOP_LOGINFO_CODE_TASK_TERMINATED rather than the non-sensical 0x1CA86D0. Changed 30050000 to 0x30050000 in _base_sas_log_info.
- CSET CQ262005: Turning on the command line option initialize_target_mode is resulting in driver panic under 32 bit Linux OS. The panic is due to a reference to ioc->pfacts when its set to NULL. Normally ioc->pfacts is set to valid pointer, however in this case it was set to NULL. It got set to NULL when the diag reset was issued as part of the initialize_target_mode handling. The reason it was set to zero is due to a bug in _base_get_ioc_facts. From _base_get_ioc_facts we are refreshing the IOC FACTs data. In that code before we refresh this structure, we are initializing the structure to zero. When it is getting initialized to zero, the size passed into the memset call was too large. The size was set to 64 bytes, when it should of been 60 bytes. This resulted in the parameter ioc->pfacts getting clear.

Major Changes For Version 13.00.00.00-1
Release Date: 02/17/2012

General Changes

- CQ 252247: Update version number to 13.00.00.00-1 to indicate this driver as a Phase-13 GCA release driver

Defect fixes

- None.

Major Changes For Version 12.254.03.00-1
Release Date: 02/15/2012

General Changes

- None.

Defect fixes

- CSET CQ251542 (CQ251237): A hard drive is being OFFLINED when there is a hard reset issued in combination with hot removing another hard drive. Following host reset, firmware discovery that is reassigning another hard drive in the topology to the same device handle as the device that is getting hot removed. Until the driver device removal routine is called, there will be two hard drives with the matching device handle in the internal device link list. In the device removal routine, we call a separate function which is moving the device from BLOCKED into OFFLINE state. Since this routine is having the device handle passed in as input parameter, the routine will be traversing the internal device link list searching for matching device handle. Since there will be two devices with matching device handle, there will be both devices getting OFFLINED. To fix this issue, we should be the sas address as input instead of device handle.

Major Changes For Version 12.254.02.00-1
Release Date: 12/08/2011

General Changes

- CQ234984: Update version number to 12.254.02.00-1.

Defect fixes

- None.

Major Changes For Version 12.254.01.00-1
Release Date: 11/28/2011

General Changes

- CQ231021: bump driver version
- CSET 230227 (CQ 230140): MPI2 Update Rev U (2.0.23)
- CSET 228119 (CQ 220045): Fix security scan issues reported by source code analysis tool. Modified the source code as per the findings reported by the source code analysis tool. None of the driver functionalities has changed. Source code for the following functionalities has been touched:
  - SMP Passthrough IOCTL
  - Debug messages for MPT Replies
  - Task Management using sysfs
  - Disk removal
  - Trace Buffer
- CSET 228120 (CQ 208765): Improvement were made to better protect the sas_device, raid_device, and expander_device links. There were possible race conditions surrounding reading an object from the link list while from another context in the driver it was removing it. The nature of this enhancement is to rearrange locking so the link lists are better protected.
- CSET 228117 (CQ 220601): Add support for in the IOCTL path for opcode MPT2COMMAND. For SATA_PASSTHROUGH commands that end with time outs, the driver will issue a target reset instead of host reset.
- CSET 228118 (CQ 213206): Driver is snooping on completions for the 0x31110630 logfinfo. After the scsi mid layer sent the same command with four retries and same loginfo, the driver will return DID_NO_CONNECT instead of DID_SOFT_ERROR, and will and offline the device. This will prevent any further request from the filesystem layer.

Defect fixes

- CSET 230301 (CQ 230280): An inactive volume that is activated is not seen immediately by the OS: The events that are sent by controller firmware to driver have changed for when an inactive volume is activated. In previous firmware releases, the driver would have received a MPI2_EVENT_IR_CHANGE_RC_UNHIDE event when a volume was activated, whereas now its receiving MPI2_EVENT_IR_CHANGE_RC_REMOVED. Due to this change, the driver now is sending a target reset to the volume at the same time the Volume is getting detected by the OS. The OS is sending an INQUIRY, and its getting aborted by the background target reset, which results in the volume never getting reported. To resolve this issue, the driver is not needing to send the target reset to the volume when its MPI2_EVENT_IR_CHANGE_FLAGS_FOREIGN_CONFIG.

Major Changes For Version 12.00.00.00-1

Release Date: 11/08/2011

General Changes

- CQ 228662: Update version number to 12.00.00.00-1 to indicate this driver as a Phase-12 GCA release driver.

Defect fixes

- None.

Major Changes For Version 11.254.01.00-1
Release Date: 08/30/2011

General Changes

- CQ 208433: Correctly identify the customer boards and display branding string in the message logs.
- CQ 209775: Add support for SATA Initialization Timeout:
  When the driver sends internal TUR to check the state of the hotplugged device if the command is returned with non zero IOC status then the command is retried until the retry count is exhausted. To handle the drives which are in SATA Initialization failure state the driver checks the loginfo for 0x31111000 (which is SATA Initialization Timeout code) and if so, the driver retry the TUR once (instead of retry count times) to check if the device is ready after the target reset initiated by the firmware, and if the driver receives the same 0x31111000, it gives up adding the device.
- CSET 217005 (CQ 217001): Add MPI2 Rev R (2.0.12) header files (for Phase 12.0).
- CQ 218153: Change driver version to 11.254.01.00 for phase 12 pre alpha release.

Defect fixes

- None.

Major Changes For Version 11.00.00.00-1

Release Date: 08/23/2011

General Changes

- CQ 215693:- Update version number to 11.00.00.00-1 to indicate this driver as a Phase-11 GCA release driver.

Defect fixes

- None.

Major Changes For Version 10.254.02.00-1

Release Date: 08/17/2011

General Changes

- None.

Defect fixes

- CSET 213015(CQ 208648): IOCTL request are taking a long time to complete back from device driver. It appears the application has hung: 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 10.254.01.00-2

Release Date: 07/06/2011

General Changes

- None.

Defect fixes

- CQ 204696:- Booting to SLES 9 SP4 after DUD installation is getting failed - This was regression of CQ 166870. As part of that the script files in the source code are appended with .sh name, so the clean script is renamed from clean to clean.sh. But the DUD creation script for SLES9 (make_dud) is not updated to call clean.sh instead of clean. This caused linking on .o files compiled for default kernel to create .ko file for smp kernel and hence resulted in corrupted memory mapped module, which on load resulted in panic.

Major Changes For Version 10.254.01.00-1

Release Date: 06/06/2011

General Changes

- CQ 199890:- Adding branding support to identify customer specific adapter.

Defect fixes

- CSET 200702(CQ 187701): Port Reset is taking long time(around 5 mins) to complete while issued during creating a volume : 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. It is okay if the driver returns with error from slave_configure becuase from the post reset handling, the driver is scanning for new volumes, and the volume is eventually added.

Major Changes For Version 10.00.00.00-1

Release Date: 05/11/2011

General Changes

- CQ 194609:- Update version number to 10.00.00.00-1 to indicate this driver as a Phase-10 GCA release driver.

Defect fixes

- None.

Major Changes For Version 09.254.02.00-1

Release Date: 04/26/2011

General Changes

- None.

Defect fixes

- CSET 192375(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.
- CSET 192374(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.
- CSET 192370(CQ 178865) - 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 a 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 owned the callback index due to the timeout. With this change, this will insure the RELEASE will be sent.

- CSET 186524(CQ 184414) - Hang when there is smart error on IBM platform: Driver was a sending a SEP request during interrupt context which required sleeping. 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.
- CSET 186520(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.
- CSET 186518(CQ 184822) - Update MPI2 headers files for fix bug in IO Unit Page 10.

Major Changes For Version 09.254.01.00-1

Release Date: 03/07/2011

General Changes

- CQ 180565 – MPI2 Rev R (2.0.10) and v2.00.19 header files (for Phase 10.0) are added.
- CSET 182617(CQ 181789) - Add support for Customer specific Identification: Report branding messages when device driver loads, based on specific customer subsystem vendor and device Ids.

Defect fixes

- None.

Major Changes For Version 09.00.00.00-1

Release Date: 02/21/2011

General Changes

- CQ 179053:- Update version number to 09.00.00.00-1 to indicate this driver as a Phase-9 GCA release driver.

Defect fixes

- None.

Major Changes For Version 08.254.04.00-1

Release Date: 02/11/2011

General Changes

- CQ 177330:- Adding support for displaying customer branding information for the Mustang controller.

Defect fixes

- None.

Major Changes For Version 08.254.03.00-1

Release Date: 01/18/2011

General Changes

- None.

Defect fixes

- CQ 171768:- RHEL4 Driver Update Disks are not recognizing the Thunderbolt and Mustang controller. To fix this corresponding PCI IDs are added in driver update disk configuration files.
- CQ 171754:- The shutdown/reboot is not completed and stuck in driver shutdown handler: The change done in CQ 166865 is causing this and it is due to some lock contention in calling the scsi_remove_host from the shutdown handler. The scsi_remove_device also blocked. The shutdown handler is modified so that it will neither remove host nor remove devices.

Major Changes For Version 08.254.02.00-1

Release Date: 11/22/2010

General Changes

- CQ 166875:-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)
- CQ 166865:- Block I/O when system shutdown is in process. 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 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 remove the devices from the OS prior to freeing its memory.
- CQ 166026:- Phase 9 revision Q header update
- CQ 160686:- 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.

Defect fixes

- CSET 166867(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 08.00.00.00-1

Release Date: 10/29/2010

General Changes

- CQ 148653:- Update version number to 08.00.00.00-1 to indicate this driver as a Phase-8 GCA release driver.

Defect fixes

- None.

Major Changes For Version 07.254.08.00-1

Release Date: 10/26/2010

General Changes

- None.

Defect fixes

- CQ 161421 - Removal of driver & adding the driver multiple times on RHEL 4.8 OS causing kernel panic.: The driver was releasing the reference count of the scsi_host structure and then trying to access ioc which is actually private data of the scsi_host structure and hence resulting in panic. This is fixed by moving the release of reference count to the end of the remove handler.

Major Changes For Version 07.254.07.00-1

Release Date: 10/01/2010

General Changes

- CSET 158279(CQ 155831) - Add support for Customer specific Identification: Report branding messages when device driver loads, based on specific customer subsystem vendor and device Ids.

Defect fixes

- CSET 158283(CQ 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.
- CSET 158281(CQ 144099) - False timeout after hard resets: There were two issues which lead 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 notifications are not handled due to events ignored while the broadcast asyn event is activity 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 example are clear reservations, microcode download, and mode select.

Major Changes For Version 07.254.05.00-1

Release Date: 08/23/2010

General Changes

- CQ 151553 - 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.

Defect fixes

- CSET 151561(CQ 148348) - Increasing max_queue_depth in Linux Driver not honored above 30,000 - Bug fix in the queue depth calculation when queue depth exceeded the MaxReplyDescriptorPostQueueDepth. The algorithm was resizing the queues incorrectly, that is fixed in the driver.

- CSET 151560(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.
- CSET 151556(CQ 141553) - When zoning end devices, the driver is not sending device removal handshake algorithm to firmware. This result in controller firmware not sending sas_topology_add events when the device is added next time. The current design is avoiding the handshake when the VACANT bit is set in the PHY status. 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.

Major Changes For Version 07.00.00.00-1

Release Date: 07/30/2010

General Changes

- CQ 148653:- Update version number to 07.00.00.00-1 to indicate this driver as a Phase-7 GCA release driver.

Defect fixes

- None

Major Changes For Version 07.254.03.00-1

Release Date: 07/06/2010

General Changes

- None

Defect fixes

- CSET 143487(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 of the SCSI state and status 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 and status transitions back to good values, the serial number check will work, and the device will not be deleted
- CSET 143481 (CQ 136939) - Bad request seen after many hours in reset expander test: The firmware development 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 a 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 addition 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

Major Changes For Version 07.254.02.00-1

Release Date: 05/24/2010

General Changes

- CQ 141004 - 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 139813 - MPI2 Rev O (2.0.7) and v2.00.16 header files (for Phase 7.0)
- CQ 139811 - 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 139805 - 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 requests.

Defect fixes

- CSET 139807(CQ 134854) - Fix panic in the driver interrupt handler 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.

Major Changes For Version 06.00.00.00-1

Release Date: 05/06/2010

General Changes

- CQ 137376– Updated version number to 06.00.00.00-1 to indicate this release is a GCA release.

Major Changes For Version 05.254.05.00-1

Release Date: 05/03/2010

Defect fixes

- CSET 136629(CQ 133558) - While running heavy IOs and abort tasks through IOCTL interface every 7 seconds on a random drive, I/O errors were seen on the drives and I/Os failed: The bug is in the driver is due to not clearing the per device tm_busy flag following the Task Mangement 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 05.254.04.00-1

Release Date: 04/23/2010

General Changes

- CQ 124059 -- Drive S/N check for SATA drives: Add support to obtain the serial number for all devices, it will be 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, and then add the new.
  .

Defect fixes

- CSET 135091(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.

- CSET 132889(CQ 132867) –When a user invoked HBA scan through the sysfs entry "/sys/class/scsi_host/hostX/scan" the SML scans for devices up to the maximum number of channels supported by the LLD for the particular controller and in each channel the scan goes to the maximum number of targets supported. The mpt2sasbtm driver sets the max_channel value to default (0) one channel and the max_id (the maximum number of targets) to -1. This was done to allow the host to scan the maximum possible number of targets. Since the -1 equals to 4294967295(0xFFFFFFFF) in unsigned, the scan was continued until the target ID reaches it. So the user sees it as a console freeze. For all non configured targets the driver throws a warning saying the sas_device=NULL, this print adds delay for scanning. This is fixed by setting the max_id to a small number which is equivalent to the number of targets and volumes supported by a particular controller (retrieved from IOC facts) and by moving the message prints from default to debug only prints.

Major Changes For Version 05.254.03.00-1

Release Date: 04/06/2010

General Changes

- CSET 131401(CQ 131193) – Added support for internal TUR retries until the command retry count is exhausted while handling the Link status change link up events.
- CSET 131085(CQ 125138) - Add additional sas_address/phy messages in driver.
  Adding additional messages to the error escalation callbacks which display 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.

Defect fixes

- CSET 131400(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.

- CSET 131091(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.
- CSET 131089(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.
- CSET 131083(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.
- CSET 131081(CQ 130633) - Volume not deleted after host reset
  BUG DESCRIPTION:
  This test case involves creating two RAID1 volumes, and 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 or optimal, or degraded.
So for volumes that have a status of missing or failed, the driver will mark them for deletion.

Major Changes For Version 05.254.02.00-1

Release Date: 03/31/2010

Defect fixes

- CSET 130576(CQ 125906) - Mpt2sas and blocked devices
  Bug Description:
  If error recovery is occurring at the same time when a device is blocked, the escalation will always 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.
- CSET 130574(CQ 125307) (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.
- CSET 130572(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.
- CSET 130141(CQ 129053) - Data corruption after drive-pull during diagnostic reset.

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:
(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 hot-plug work task 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 hot-plug work task was activated to report all missing devices to the scsi mid layer.

Major Changes For Version 05.254.01.00-1

Release Date: 02/22/2010

General Changes

- CQ 124294 – 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 124086 – The changes suggested by the external code review are implemented
- CQ 124085 – The driver is modified to use default descriptor instead of scsiio descriptor while sending RAID passthru commands, this is as per latest MPI spec. This is a Must requirement for RAID passthru to work with FW versions later than 05.00.00.00
- CQ 124078 – The driver is modified to detect a drive which returns sense data 04/19/01 (sense/asc/ascq) during the hotplug time,

- CQ 124071 – Added three new shost attributes: host_trace_buffer, host_trace_buffer_enable, and host_trace_buffer_size for registering and retrieving firmware trace buffer through the sysfs interface.
  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 atttribute contains a maximum 4KB window of the buffer.
  In order to read the entire host buffer, the offset has to be written 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.
- CQ 124065 – In OS distributions KERN_DEBUG logging turned off by default and it is painful to configure systems so they receive KERN_DEBUG messages when debugging issues. Many times when looking at logs, most critical information is missing. So all the KERN_DEBUG messages are converted to KERN_INFO.
- CQ 124063 – MPI headers updated to revision N
- CQ 124061 – A new sysfs attribute ioc_reset_count is added, which can retrieved by reading the /sys/class/scsi_host/hostX/ioc_reset_count. This will provide the number of hard and soft reset issued by the driver.
- CQ 124058 – Added support to disable topology discovery during driver load time, a new module parameter disable_discovery is added and if it is set 1 during driver load time then the driver will not issue port enable call to the firmware and hence will not kick off the discovery. At a later point in time user can start the discovery by issuing host reset either through sysfs or ioctl interface

Defect fixes

- CSET 124356(CQ 119210) – Kernel panic during early boot: The panic is in _scsih_sas_device_remove when the driver was attempting to delete an object from the sas_device link list when the object was not present. This is fixed by adding a check to verify the whether the object is present in the list before deleting it from the list.
- CSET 124089(CQ 118104) - HBA firmware fault 600F: 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.
  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.

- CSET 124068(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 configuration page timeout. Since there was a host reset active, the driver would return a FAILED response to the SML. Then the SML would begin 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.

Major Changes For Version 05.00.00.00-1

Release Date: 02/09/2010

General Changes

- CQ 120990– Updated version number to 05.00.00.00-1 to indicate this release is a GCA release.

Major Changes For Version 04.254.05.00-1

Release Date: 01/25/2010

General Changes

- CQ 118212 - Updated copyright to year 2010.

Defect fixes

- CSET 119658(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.

- CSET 118213(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 IO when firmware is sending a device reset. This is fixed by modifying the logical expression to evaluate properly.
- CSET 117577(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.
- CSET 117576 (CQ 113732) - 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.
- CSET 117573(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 hot-plug 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 hot-plug work threads, and any pending work. Also added were several sanity check for the flag "ioc->remove_host".

Major Changes For Version 04.254.04.00-1

Release Date: 12/21/2009

Defect fixes

- CQ 114381 – Direct attached drives are not shown if the controller is configured in enclosure slot mapping: The previous version of thedriver was modified to check for zero NumSlots, Due to coding error, the zero check is implemented as non-zero check and resulted in not adding a valid enclosure. This is fixed in this release.

Major Changes For Version 04.254.03.00-1

Release Date: 12/18/2009

Defect fixes

- CSET 114233(CQ 106189) – Bus/target mapping in enclosure/slot mode and with a multi-path configuration was becoming corrupted because the mapping table only had the enclosure logical ID and did not have the PhyBits information. With multi-pathing, an enclosure instance is the combination of the enclosure logical ID and the PhyBits field (which PHYs are used to connect to the enclosure).The fix was to add a PhyBits field to each Mapping Table entry and modify the mapping code to check both the enclosure logical ID and the PhyBits when trying to match an enclosure add event to possible existing entries in the mapping table. Code was added in each routine that adds mapping table entries to also add the PhyBits.Also, added a check when processing an enclosure add event for NumSlots equal to 0. This should never happen since the enclosure should at least have an SES device and it needs at least one slot in the enclosure so the SES device can be mapped. If NumSlots = 0 then the driver does not add the enclosure to the enclosure table and a error message will be logged.
- CSET 113899(CQ 113882) – The timestamp is wrongly calculated in 32bit OS due to improper type casting. That is fixed in this release
- CSET 113897(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.
- CSET 113896(CQ 107271) – When a drive with write cache enabled is removed then the driver stops further processing of events like device deletions or insertions. The system needs reboot to recover. This had happened due to the code change to fix the issue CSET 103586 in the 4.00.01.00 driver. The driver code is modified to unblock the device before removing the device. This enables the synchronize cache command to be processed and avoids the deadlock and allows further processing of the events.

- CSET 113894(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.

Major Changes For Version 04.254.02.00-1

Release Date: 11/26/2009

General Changes

- CSET 111075(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.

Defect fixes

- CSET 111081(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.
- CSET 111079(CQ 110306) - ppc64 sas2flsh failure investigation: On ppc64, a 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.

- CSET 111077(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.

Major Changes For Version 04.254.01.00-1

Release Date: 11/06/2009

General Changes

- CSET 106328(CQ 106191) – MPI Headers updated to version 02.00.14.
- CSET 103636(CQ 103188) – Added RHEL4.7 back into the dkms prebuilt binaries in addition to the existing rhel4.8 binaries.
- CSET 102609(CQ 098428) – Add new command line parameter sdev_queue_depth. This will globally set the queue depth for all SAS devices at once (ignoring SATA and volumes). This parameter can be set at driver load time.

Defect fixes

- CSET 107179(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.
- CSET 107177(CQ 104563) – Endian fixes.
- CSET 106127(CQ 105334) – The driver logs error indicating the enclosure can not be removed as it is already removed if one of the path of multipath capable enclosure is connected and removed: On removal of an enclosure the PhyBits reported in the SAS Enclosure Status Change event are set to 0. The driver was using the Logical ID and PhyBits to locate the enclosure in the driver enclosure table. This would not find a match due to the 0 PhyBits. Changed the enclosure removal handler function to use the

enclosure handle to find the entry on a removal instead of the combination of Logical ID and PhyBits.

- CSET 103607(CQ 102692) – When the dkms rpm is installed, it updates the driver properly in the /lib/modules and it recreated initrd image. But the resulting initrd image does not contain the mpt2sasbtm driver due to the reason that dkms rpm does not add an entry for mpt2sasbtm in /etc/modprobe.conf. Hence if the OS is installed using the DUD created using dkms, after installation while doing the reboot the drivers are not getting loaded and resulted in panic. To fix this the "MODULES_CONF_ALIAS_TYPE[0] = "scsi_hostadapter" " line is added in dkms.conf file to let the dkms add an entry in the modprobe.conf. This makes the mkinitrd to add the driver in the initrd image.
- CSET 103588(CQ 102810) – system halts during jammer scripts and medusa stress: There is a kernel 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.
- CQ 102488 – When expanders are connected in topology a kern error "failure at mpt2sas_scsih.c:5331/_scsih_add_device ()!" is logged: The link status change event handling, mistakenly tried to add expanders in the topology as end device hence the failure is reported. Fixed by adding proper break statement in switch case of link status change handling.
- CSET 102476(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 04.00.00.00-1

Release Date: 10/12/2009

General Changes

- CQ 102472– Updated version number to 04.00.00.00-1 to indicate this release is a GCA release.

Major Changes For Version 03.254.04.00-1

Release Date: 09/24/2009

General Changes

- CSET 99576(CQ 99447) – Add support for F/W multi-pathing: Host mapping code was modified to look at the PhyBits field in the SAS Enclosure Status Change event. The Phybits indicate which phys are being used for this enclosure. The PhyBits are maintained in the local driver enclosure table as well as in the DPM Page 0 persistency page entries. The use of the PhyBits allows the driver to determine the multi-pathing capabilities when dual ported devices reside in the same Enclosure Logical ID (attached to two different expanders).
- CSET 95346(CQ 64808) – Added 'fixid' rpms.

Defect fixes

- CQ 99232 – Linux Driver-SAS Volume mapping failure while creating RAID volumes in PPC systems. The endianess conversion was missed in couple of places in mapping code which is causing the issue. This is fixed by addressing those conversions.
- CQ 99675 –Persistent SAS address creation for ATAPI device failed: When an ATAPI device is connected in the topology, it is not visible to the OS and the driver throws error indicating the SAS address creation failed: The driver is modified to issue Identify packet device command (0xA1) for ATAPI devices and Identify device (0xEC) for other devices. Also the SASStatus is checked for success of the command in addition to IOC status. In addition the driver is modified to use the SAS device Page 0 provided SAS address if the Hashed SAS address creation is failed. This change will allow the user to see the device even if the persistence SAS address creation is failed. But the IDs may not be persistent.
- CSET 99578(CQ 98589) – Snowmass - IO Stress and random TM causes STAF script to lock up: This was due to dead lock between the user issued task management and kernel error recovery handler issued task management. This is fixed by moving the task management mutex to the core function which issues task management.

Major Changes For Version 03.254.03.00-1

Release Date: 09/09/2009

General Changes

- CQ 95350 – Binary level support for PPC architecture is added
- CSET 97581(CQ 97308) - 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.
- CSET 95625(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.
- CSET 95621(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.
- CSET 95334 (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 as missing or it will deadlock in the mid layer.
- CSET 93303 (CQ 90767) - 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.

Defect fixes

- CQ 98244 – Kernel panic when the device_rescan script is executed multiple times without enough time in between. Then the slave_destory functions ends up accessing a NULL sas_device pointer. This is fixed by checking for NULL before accessing the pointer.
- CQ 95345 – Fix SLES9 SP4 DUD. Modified the dud update.post script not to do chroot and also modified the makedud script to clean the previous built object files before making a new kernel build.
- CSET 97578(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.
- CSET 97456(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.
- CSET 96917 (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 it is already in __le32 format.
- CSET 95341(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.

- CSET 95341(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 un-load and reboot context.
- CSET 95339(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.
- CSET 95337(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.
- CSET 95330(CQ 94826) -LSI SAS Driver Stop code during W2K3E SP2 R2 X86 OS Installation. This issue is found in Windows but applicable to Linux also. The adapter being used in this test had a full persistency table (due to a prior F/W issue). The full persistency table uncovered a driver bug where a Bad Index value (0xFFFF) was being used to index into the local copy of the persistency table, resulting in an invalid memory access and blue screen. A mapping table entry can have a Bad Index value if the persistency table is full and that entry cannot be added to the persistency table. The driver fix is to check for a Bad Index value and not attempt any operations on the persistency table for that entry.
- CSET 95220(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.

- CSET 94409(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.
- CSET 92740 (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.
- CSET 92783(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.

Major Changes For Version 03.254.01.00-1

Release Date: 08/10/2009

General Changes

- CQ 82978 - SLES9 SP4 binary level support is added for x86 and x86_64 architectures
- CQ 82978 - Support for diskdump is added.
- CSET 66952 (CQ 37567)- Support for stopping driver when Firmware encounters fault. Added command line option 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.
- CSET 90009 (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.
- CSET 90006 (CQ 35970) - Create SYSFS SHOST attribute for sending task management request. Created 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 are the supported numbers:

1 = diag reset
2 = message unit reset
3 = abort task
4 = target reset
5 = lun reset
6 = abort task set

- CSET 90003 (CQ 14691) - Add support for RAID Action System Shutdown Initiated at OS Shutdown:
- CSET 90017 (CQ 89677) - Linux SAS2.0 - Adding Phase 4.0 MPI Headers - revision L
- CSET 90015 (CQ 89640) - SAS2208/Thunderbolt support: Add support for PCI Device ID's range {0x80 - 0x87}
- CSET 82973 (CQ 82588) - Add support for sending multiple task management request at the same time
- CSET 90032 (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 do the device removal handshake protocol.

Defect fixes

- CQ 68900 - In device persistence mapping mode with RHEL4/SLES9 driver if an SATA disk is moved between different PHY's then the Target ID are changing. The expected behavior is the ID should not be changed even if the device is moved to a different PHY. This is fixed by creating a unique ID for SATA devices if the mapping mode is device persistence mapping. The ID is generated based on the SerialNumber and ModelNumber of the SATA drive and that ID will be used as Identifier for SATA device mapping instead of the SAS address.
- CQ 90317 - Linux Driver mpt2sasbtm : The driver persistency mapping is not working for Volumes. If two volumes are created and the volume with ID0 is deleted then rhe expectation of the persistence mapping is, the volume with ID1 should be retained with ID1 and any newly created volume should be assigned with ID0. But the behavior observed is, the

old volume gets ID0 and the newly created volume gets ID1. The driver is modified to do proper sized memory copy for the DPM page read from NVRAM to fix this problem.

- CSET 90030 (CQ 85944) - Corruption caused by SAS Truncated CMD Frame: Added 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.
- CSET 90021 (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.
- CSET 90019 (CQ 80574) - 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 requests using the wait_for_completion_timeout API. The previous implementation was only calling init_completion at driver load time.
- CSET 90026 (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.
- CSET 90023 (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.

- CSET 90028 (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.

Major Changes For Version 02.00.01.00-1

Release Date: 06/29/2009

General Changes

- None.

Defect fixes

- CQ 83973 – Volume created runtime with any tools like SAS2IRCU is recognized only after reboot: When the firmware sends a IR configuration change event the driver will be doing a match of the existing map table with the WWID of all the volumes added/created to find out the free mapping slot for the newly created/added volume. Due to an erroneous indexing even the physical disks events are considered for matching (with WWID == 0) and hence there was no space in the mapping table for newly created volume. The indexing is fixed properly and additional checking is added to avoid all Non-Volume events.

Major Changes For Version 02.00.00.00-1

Release Date: 06/18/2009

General Changes

- CQ 81655 – Updated version number to 02.00.00.00-1 to indicate this release is a GCA release.

Defect fixes

- None.

Major Changes For Version 01.254.05.00-1

Release Date: 06/11/2009

General Changes

- CQ 81638 – Removed the Target Mode Driver Sources from the source tarball

Defect fixes

- CSET 82303 (CQ 81462) – Ported the fixes done in mpt2sas driver for proper power management support.

Major Changes For Version 01.254.03.00-1

Release Date: 05/19/2009

General Changes

- CQ 78716 - Support for RHEL4 Update 8 is added. The RPM contains modules for 2.6.9-89 kernel. The dkms prebuilt binaries are replaced with the 2.6.9-89 kernel based binaries.
- CQ 75566 - A new proc entry /proc/mpt2sasbtm/version is added to provide the driver version.

Defect fixes

- CQ 78884 - The dkms spec file has wrong kernel names for smp, largesmp and hugemem flavors and hence the dkms binary based installation will not install the .ko files for smp, bigsmp and hugemem kernels. Modified the kernel names in spec file to have proper flavors.
- CQ 75583 – Deleting volumes and multiple reboots causes panic. While updating the missing count for IR volumes the driver didn't consider the reserved ID (ID 0) in loop count and hence when two volumes are missing the missing count is updated for three entries in map table and the third entry in the map table is the disk present in the first slot of enclosure. The missing count will get incremented for every boot and hence after every boot the DPM is getting added with same map table entry. Because of this after 10 reboots (max enclosures is 10) the driver panics. This is fixed by updating the volume missing count update loop to start from actual volume mapping start index instead of 0.
- CSET 76186 (CQ 75657) - IR volumes deleted and re-created outside the OS do not get mapped on first boot: Fix for IR volumes: Whenever adding an IR volume and there is no space available in the mapping table, look at

all subsequent Volume Add entries in the IR event list and see if any match the Volume ID of the first IR mapping entry. If not, reuse that entry. If so, move to the next available IR mapping entry and repeat the Volume ID check. Fix for bare targets: At start-of-day, if any of the target additions cannot be mapped due to no space available, set a flag that a device add failed. After all initial mapping events have been processed, do the start-of-day checks (to see the missing counts), force a write of all changed DPM Page 0 entries, do a MUR or diag reset (MUR if events can be replayed), then retry the controller initialization (one more time). This allows the driver to determine what devices that were specified in the persistent mapping data are now missing, so those mapping entries can be reused for newly added devices.

- CSET 78707 (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, and 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 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.
- CSET 78708 (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 78709 (CQ 75081): (1) add support to retry READ_CAPACITY16 when there is a unit attention.
- CSET 78710 (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
- CSET 78711 (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.

- CSET 78712 (CQ 75917)- added sanity checks on volume add and removal to ignore events for foreign volumes.
- CSET 78713 (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.
- CSET 78714 (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.

Major Changes For Version 01.254.02.00-1

Release Date: 04/15/2009

General Changes

- This driver is functionally equivalent to the mpt2sas driver version 01.255.02.00-1 (driver for RHEL5/SLES10/SLES11) without the following major features included.
  - Linux SAS transport layer
  - Access to hidden physical disks in IR volumes
  - MSI-X based interrupt handling support
  - Kernel specific features like Linux EEDP API support

Major Changes For Version 00.254.12.00-1

Release Date: 03/23/2009

General Changes

- Initial release.
- This driver is functionally equivalent to the mpt2sas driver version 00.255.12.00-1 (driver for RHEL5/SLES10/SLES11) without the following major features included.
  - Linux SAS transport layer
  - Access to hidden physical disks in IR volumes
  - MSI-X based interrupt handling support
- Implemented Host Mapping (Bus/Target ID mapping to device handle) at the driver level.