CAM-based ATA implementation
3 ... 2 ... 1 ... Lift-off!
Alexander Motin
mav@FreeBSD.org
• ATA(4) was started more than 10 years ago and handled generic ATA controllers well. But now there are problems:
  – modern controllers use different APIs - driver API was fuzzy and modern controllers still don’t fit it well;
  – modern controllers support command queues - no support;
  – SATA controllers and disks support NCQ - no support;
  – SATA 2.x controllers support Port Multipliers - very limited support;
  – some with FIS-based switching - no support;
  – SATA has number of additional features, such as interface power management - no support;
  – ATAPI tunnels SCSI commands over ATA - ata(4) ATAPI drivers produce code duplication, atapicam(4) is not perfect;
  – implements unfair scheduling;
  – error recovery is quite limited due to limited state machine.
• CAM doesn’t have some of problems (command queueing, fair scheduling, state machine). For others it allows solutions.

• CAM(4) improvements:
  – XPT level was split and partially virtualized to allow support for different transport types; SPI specific code moved to SCSI XPT (thanks to Scott Long);
  – new ATA XPT implements specific management for PATA and SATA buses, including port multipliers support;
  – new XPT_ATA_IO request type allows transporting ATA protocol commands; ATAPI devices can handle both XPT_ATA_IO and XPT_SCSI_IO requests;
  – peripheral drivers do not depend on device transport (SPI, SAS, SATA, ...), only command protocol (SCSI, ATA, ...); for ATA protocol disks implemented new driver ada(4);
  – XPT code was improved to allow support more complicated bus topologies;
Updated CAM(4) structure

- **GEOM**
  - daX
  - adaX
  - cdX

- **User-level**
  - saX
  - pmpX
  - sesX
  - passX

**XPT**
Transport: Command queues. Generic error handling.

- SCSI (SPI, ...)
- ATA (PATA, SATA)

**Periph drivers.**

**Interface Modules:** Controller hardware drivers.

- aha
- umass
- ahc
- ahci
- mvs
- siis
- ata

**Bus management.**

- ISA
- USB
- PCI
- ISA
• ahci(4) driver supports:
  – integrated and add-in AHCI-compatible SATA controllers;
  – ATA and ATAPI devices;
  – each port completely independent
  – up to 32 queued commands per port;
  – NCQ;
  – SATA Port Multipliers (with FIS-based switching, when h/w supports);
  – MSI (one or multiple vectors);
  – Command Completion Coalescing (if somebody wish);
  – SATA power management;
  – I/Os of any size, up to MAXPHYS.
• mvs(4) driver supports:
  – several Marvell SATA chips: 88SX50xx, 88SX60xx and 88SX70xx, SoC;
  – ATA and ATAPI devices (ATAPI with some limitations);
  – each port completely independent;
  – NCQ (up to 31 commands per port);
  – SATA Port Multipliers (with FIS-based switching for NCQ commands, when h/w supports);
  – MSI;
  – Command Completion Coalescing (if somebody wish);
  – SATA power management;
  – I/Os of any size, up to MAXPHYS.
• siis(4) driver supports:
  – several SiliconImage SATA chips: SiI3124 - 4-port PCI-X, SiI3132/3531 - 2/1-port PCIe x1;
  – ATA and ATAPI devices;
  – each port completely independent;
  – 31 queued commands per port;
  – NCQ;
  – SATA Port Multipliers (with FIS-based switching);
  – MSI (works only on SiI3124);
  – device-initiated SATA power management;
  – I/Os of any size, up to MAXPHYS;
Refactored and turned into CAM SIM by `options ATA_CAM`.

ata(4) supports:
- all legacy ATA chips;
- ATA and for most chips ATAPI devices;
- no queued commands;
- no NCQ;
- no SATA Port Multipliers (require a lot of cleanup, difficult to keep compatibility with legacy mode);
- MSI supported for some controllers;
- I/Os of up to 512K size, depending on controller, respecting MAXPHYS.
• ATA(4) structure

<table>
<thead>
<tr>
<th>GEOM</th>
<th>User-level</th>
<th>CAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>atadisk</td>
<td>atapicd</td>
<td>atapifd</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Peripheral drivers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>atapist</td>
</tr>
<tr>
<td></td>
<td></td>
<td>atacore</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Command queue. NewBus. IRQ, DMA. Error handling.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>atacore</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Options ATA_CAM</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Generic ATA</th>
<th>atasii</th>
<th>ataahci</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bus management</td>
<td>B. m.</td>
<td>B. m.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ataissa</th>
<th>atacard</th>
<th>atapci</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISA</td>
<td>PCCard</td>
<td>PCI</td>
</tr>
</tbody>
</table>
• Kernel options:
  – 8-STABLE and 9-CURRENT support both old and new ATA stacks; 8-STABLE uses old stack by default, 9-CURRENT was recently switched to the new one.
  – To use new ATA stack kernel config should include:
    • device scbus
    • device da
    • device cd
    • device ...
    • device pass
    • device ahci
    • device ata
    • options ATA_CAM
    • device mvs
    • device siis
  – devices atadisk, atapicd, atapifd, atapist, atapicam, ataraid are not used by the new stack and should be removed.
• Kernel modules:
  – devices scbus, da, cd, pass, etc are parts of the cam module.
  – devices ahci, siis and mvs have own modules.
  – device ata same as before consists of number of ata...
    modules (ata, atapci, ataintel, ...).
  – `options ATA_CAM` should be present in kernel config and
    can’t be switched/loaded dynamically.
• Command equivalents:
  – atacontrol list camcontrol devlist
  – atacontrol cap camcontrol identify
  – atacontrol reinit camcontrol reset
  – atacontrol mode camcontrol negotiate
  – atacontrol spindown camcontrol idle/standby/sleep
  – atacontrol create/delete/... graid label/delete/...

• Device names equivalents:
  – adX adaY
  – acdX cdY
  – afdX daY
  – astX saY
  – To simplify migration, adX => adaY symbolic links created in /dev/, allowing to mount FSes by their old names without demanding to update /etc/fstab beforehand.
  – Mapping between names also reported during system boot.
• How it looks now (camcontrol devlist):

```bash
%atacontrol list
atacontrol: control device not found: No such file or directory
%camcontrol devlist
<Slimtype DVD A DS8A1P CA11> at scbus0 target 0 lun 0 (pass0,cd0)
<ST3250620NS 3.AEK> at scbus1 target 0 lun 0 (pass0,ada0)
<Optiarc DVD RW AD-7200S 1.0A> at scbus1 target 1 lun 0 (cd1,pass1)
<Hitachi HTS542525K9SA00 BBFOC31P> at scbus1 target 2 lun 0 (ada3,pass5)
<Port Multiplier 37261095 1706> at scbus1 target 15 lun 0 (pass2)
<OCZ-VERTEX 1.30> at scbus2 target 0 lun 0 (pass3,ada1)
<ST3250620NS 3.AEK> at scbus3 target 0 lun 0 (pass4,ada2)
```

• How it looks now (dmesg):

```bash
ada0 at ahcich0 bus 0 scbus1 target 0 lun 0
ada0: <OCZ-VERTEX 1.4> ATA-8 SATA 2.x device
ada0: Serial Number H262LML036XYSDZVG1JI
ada0: 300.000MB/s transfers (SATA 2.x, UDMA6, PIO 512bytes)
ada0: Command Queueing enabled
ada0: 61056MB (125043311 512 byte sectors: 16H 63S/T 16383C)
sc0 at atao bus 0 scbus0 target 0 lun 0
sc0: <Slimtype DVD A DS8A1P CA11> Removable CD-ROM SCSI-0 device
sc0: Serial Number 711050015684
sc0: 33.300MB/s transfers (UDMA2, ATAPI 12bytes, PIO 65534bytes)
sc0: Attempt to query device size failed: NOT READY, Medium not present
```
• How it looks now (camcontrol identify):

```
camcontrol identify ada0
pass1: <OCZ-VERTEX 1.4> ATA-8 SATA 2.x device
pass1: 300.000MB/s transfers (SATA 2.x, UDMA6, PIO 512bytes)

protocol          ATA/ATAPI-8 SATA 2.x
device model      OCZ-VERTEX
firmware revision  1.4
serial number      H262LML036XYSDZVG1JI
cylinders          16383
heads              16
sectors/track      63
sector size        logical 512, physical 512, offset 0
LBA supported      125043311 sectors
LBA48 supported    125043311 sectors
PIO supported      PIO4
DMA supported      WDMA2 UDMA6
media RPM          non-rotating

<table>
<thead>
<tr>
<th>Feature</th>
<th>Support</th>
<th>Enable</th>
<th>Value</th>
<th>Vendor</th>
</tr>
</thead>
<tbody>
<tr>
<td>read ahead</td>
<td>yes</td>
<td>no</td>
<td></td>
<td></td>
</tr>
<tr>
<td>write cache</td>
<td>yes</td>
<td>no</td>
<td></td>
<td></td>
</tr>
<tr>
<td>flush cache</td>
<td>yes</td>
<td>yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>overlap</td>
<td>no</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tagged Command Queuing (TCQ)</td>
<td>no</td>
<td>no</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Native Command Queuing (NCQ)</td>
<td>yes</td>
<td></td>
<td>32 tags</td>
<td></td>
</tr>
</tbody>
</table>
```
• Performance:
  – Number of random I/Os per second for different number of threads with legacy `ata(4)` driver and `ahci(4)`. Seagate ST3320418AS on ICH10R AHCI HBA.
• Performance:
  – Number of random I/Os per second for different number of threads. gstripe of four Seagate ST3320418AS on SiI3124 HBA with SiI3726 Port Multiplier and ICH10R HBA with and without Port Multiplier.
• As soon as ataraid(4) driver is not applicable to the new ATA stack, RAID GEOM class was implemented to handle BIOS-based software RAIDs.

• RAID GEOM follows modular design with APIs based on KOBJ and consists of such parts: core, transformation modules, metadata modules and graird(8) control tool.
  – core part handles requests and events queues, geom interoperation, etc.
  – transformation modules handle different data transformations, implementing different RAID levels. Now implemented: RAID0, RAID1, RAID1E, RAID10, SINGLE, CONCAT.
  – metadata modules handle all vendor-specific things, such as metadata formats, disks and volumes management, spare disks, etc. Now implemented: Intel, JMicron, NVIDIA, Promise (also used by AMD/ATI) and SiI.
• Special thanks to:
  – iXsystems Inc. for supporting my work;
  – many FreeBSD users for feedback and hardware donations.

• Questions?