CAM-based ATA implementation

Alexander Motin
mav@FreeBSD.org

- Generic ATA
  - Generic ATA controller - is just a continuation of 16bit ISA bus, allowing CPU directly access ATA device registers;
  - Controller is transparent and knows nothing about commands or requests;
  - PIO mode data transfer is just a direct CPU access to the device data register. ISA DMA controller does the same, but in hardware;
  - As soon as DMA uses different bus protocol, ATA has specific commands for it, different from PIO;
  - master/slave addressing is just one bit of device register, controller is unaware of it, as result, no shared bus access, no bus arbitration, devices can’t be accessed in parallel;
  - Generic PCI ATA controllers switched to BusMastering for DMA and added 32bit register access to speedup PIO. No architecture changes.

- ATA(4) was started more then 10 years ago and it was tuned to support generic ATA controllers with all their specifics:
  - controller is completely transparent and is not aware of command processing, as result, command protocol implemented in software, no any offload or queueing;
  - ATAPI adds ability to transport SCSI commands over ATA bus by adding PACKET command with specific command protocol, but keeps all ATA bus negotiation and management.
  - Late ATA adds TCQ support, but it wasn't effective due to lack of hardware support in controller. ATA(4) doesn't support it.

- SATA 1.x
  - SATA introduces new bus protocols, made to transport abstract frames - FIS (Frame Information Structure).
  - Now controller should be aware of commands (full sets of Command Registers) to generate command FIS properly and responsible for incoming FIS parsing.
  - Legacy ATA compatibility implemented via set of shadow registers. Bus protocol now completely separated from controller API.
  - Unluckily, to make transition easier, new logical bus protocol is still based on the same set of register transfers as before. No new functionality added, except new serial link control.

- SATA 1.x NCQ
  - SATA deprecates TCQ and introduces NCQ (not mandatory for 1.x). NCQ implies hardware queue handling to be implemented by controller using First Party DMA mechanism.
  - NCQ introduces two new ATA commands READ FPDMA QUEUED and WRITE FPDMA QUEUED with different completion reporting scheme and error recovery. There is no NCQ variant for ATAPI.

- AHCI
  - AHCI is a SATA-only controllers without legacy support.
  - AHCI removes PIO support. PIO transfer requests now also handled by controller using BusMastering.
  - Every SATA port operates independently, no master/slave.
  - AHCI handles queue of up to 32 commands per port in hardware. Both NCQ and regular commands could be queued, but not at the same time.
  - AHCI is now de-facto standard for on-board SATA controllers, but most of controllers also have legacy emulation mode. Emulation mode hides most of AHCI features.

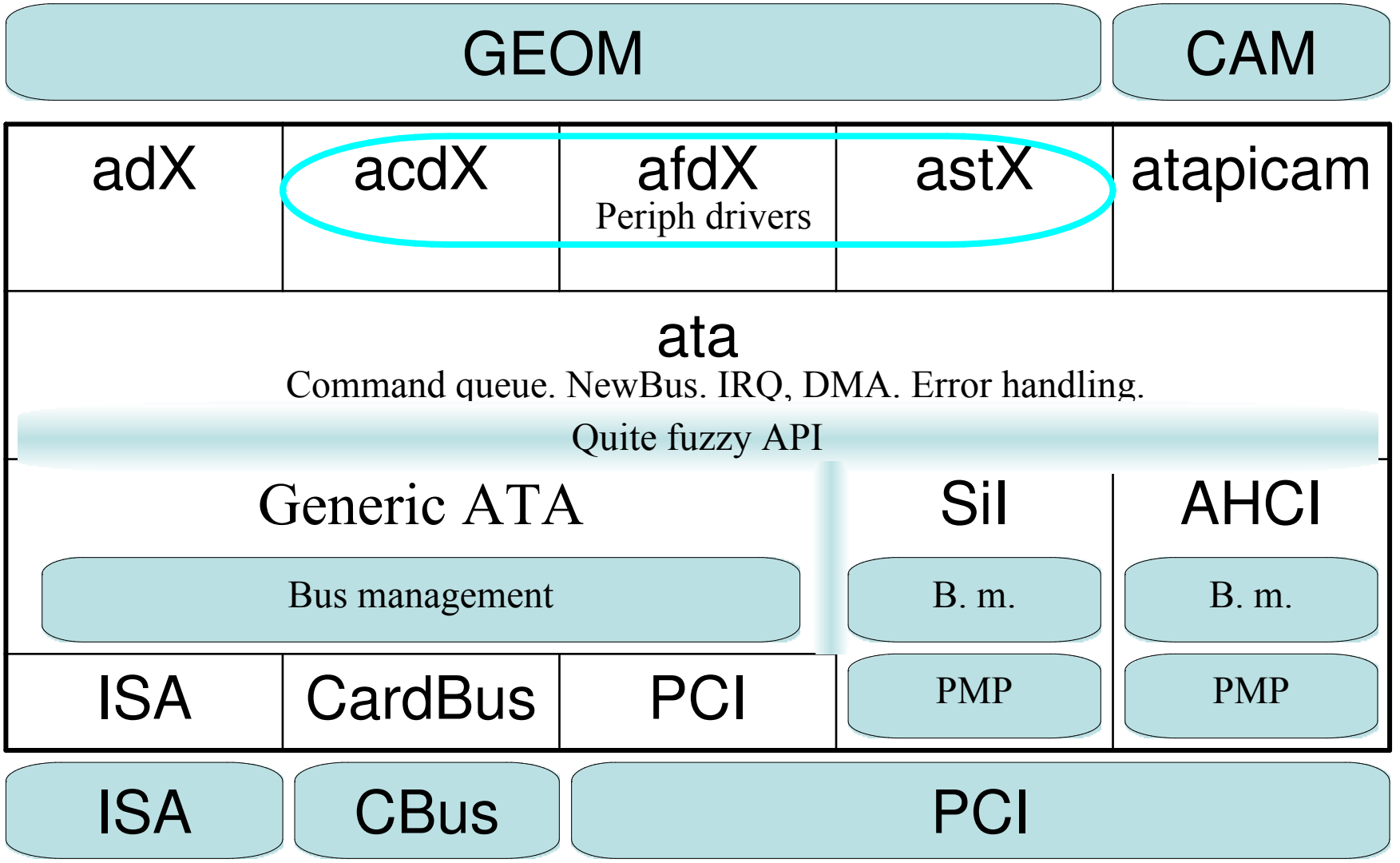
- SATA 2.x
  - This standard doubles interface speed, makes NCQ support mandatory and allows to connect up to 15 devices to single controller port, using Port Multipliers (PMP).
  - PMP works alike to VLAN-capable Ethernet switch, distributing FISes using 4bit field in FIS header on the way down and populating it on the way up.
  - To properly fill/parse that field, HBA should have PMP support.
  - Initiall AHCI unable to track status of several drives beyond PMP, so they can't work in parallel, even using NCQ.
  - FIS-based switching capability was introduced at AHCI 1.2, to address that issue, but none of existing AHCI HBAs support it yet.
  - SATA 2.x SiliconImage HBAs have own API and support FIS-based switching to effectively use PMP.

- Few words about SAS:
  - SAS electrically compatible SATA,
  - SAS uses different protocol, based on SCSI commands,
  - SAS allows port bundling to reach higher bandwidth,
  - SAS supports much more devices, using Expanders. Expanders could cascade.
  - SAS HBAs could emulate SATA protocol to support SATA drives,
  - SAS Expanders could tunnel SATA protocol to support SATA drives.

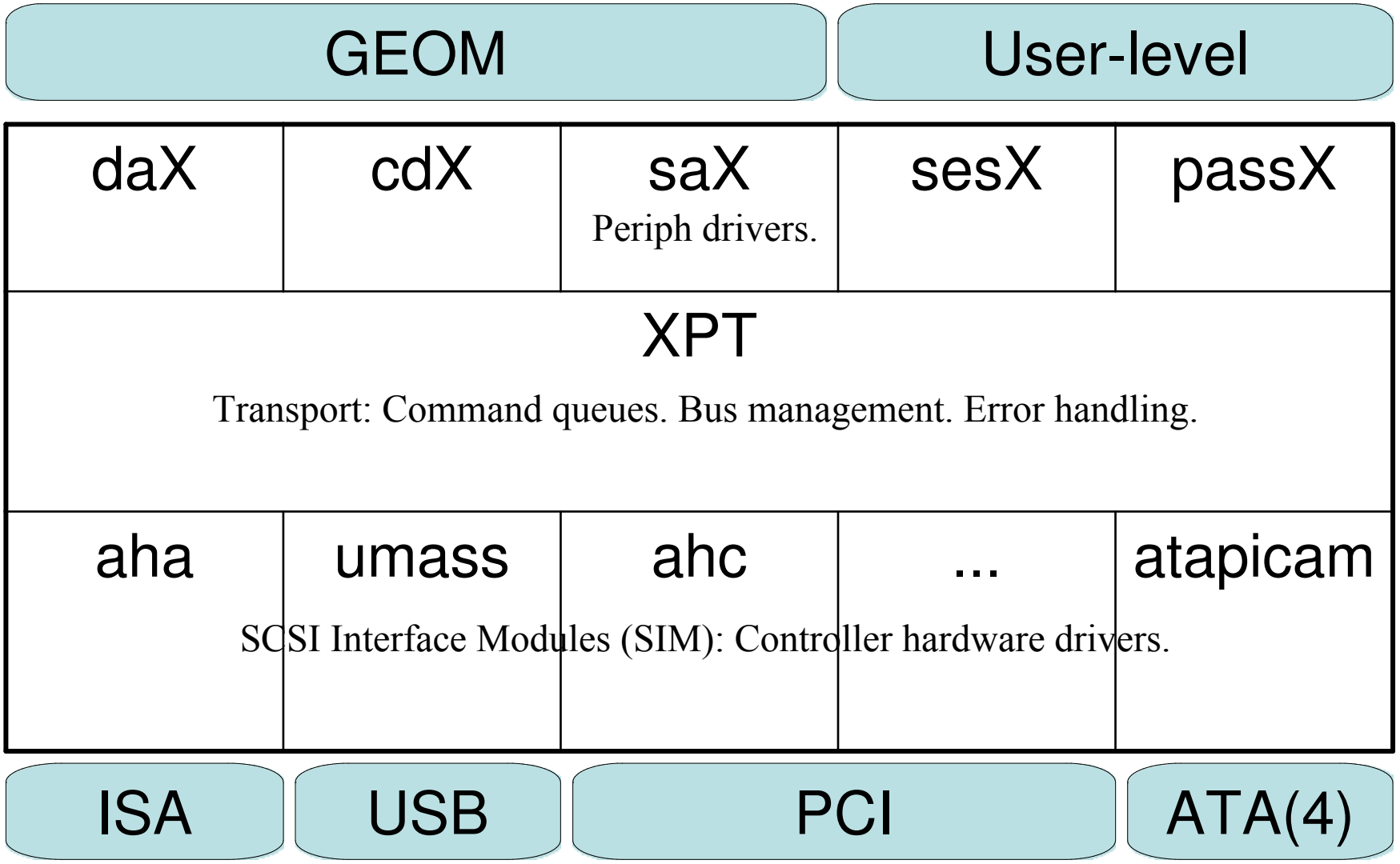
- Main technology improvements:
  - new controllers use completely different API,
  - SATA controllers support command queueing,
  - SATA controllers and disks support NCQ,
  - SATA 2.x controllers support PMP and FIS-based switching,
  - ATAPI is able to tunnel SCSI commands over ATA.
  - SAS controllers support SATA protocol, allowing SATA disks to be connected,
  - SAS Expanders could tunnel SATA protocol over SAS,

- Most of this features are not supported by ATA(4).

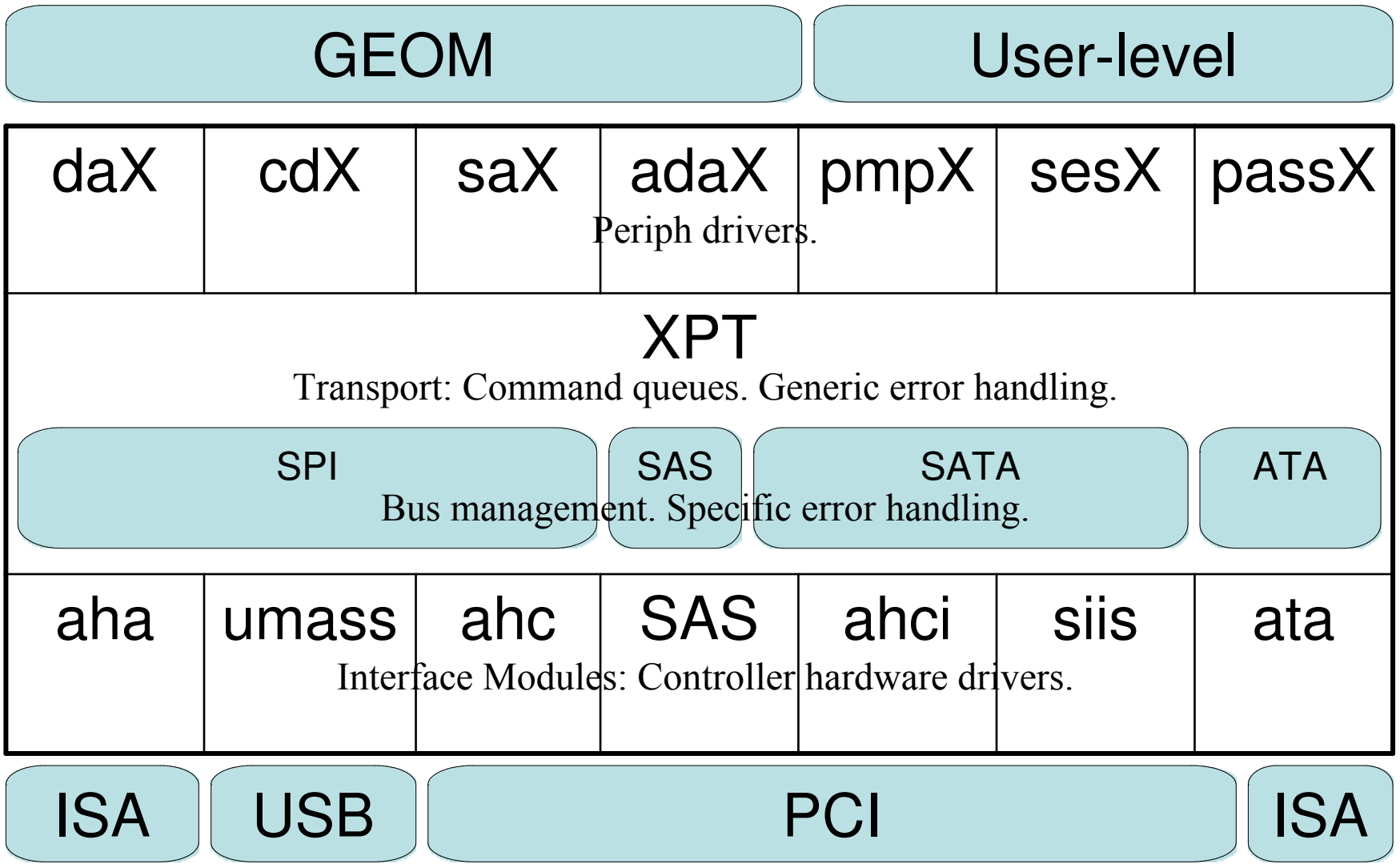
- ATA(4) structure



- Original CAM(4) structure



- Updated CAM(4) structure



- ahci(4) supports:
  - most of modern on-board, and some add-in SATAs,
  - up to 32 commands queue,
  - NCQ,
  - PMP (without FIS-based switching yet, no hardware),
  - MSI (one or multiple vectors),
  - Command Completion Coalescing (if somebody wish),
  - effective SATA power management,
  - I/Os of any size, up to MAXPHYS

- siis(4) supports:
  - several chips (3124 - fast PCI-X, 3132/3531 - moderate PCIe x1)
  - up to 31 queued commands
  - NCQ
  - PMP (_with_ FIS-based switching)
  - MSI not working for some reason for me now
  - minimal SATA power management
  - I/Os of any size, up to MAXPHYS

- wrapped ata(4) supports:
  - legacy chips,
  - no queued commands,
  - no NCQ,
  - no PMP (require a lot of cleanup, difficult to keep compat),
  - MSI supported for some controllers,
  - minimal SATA power management,
  - I/Os up to 64/128K

- How it looks now (device list):

```
%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)
```

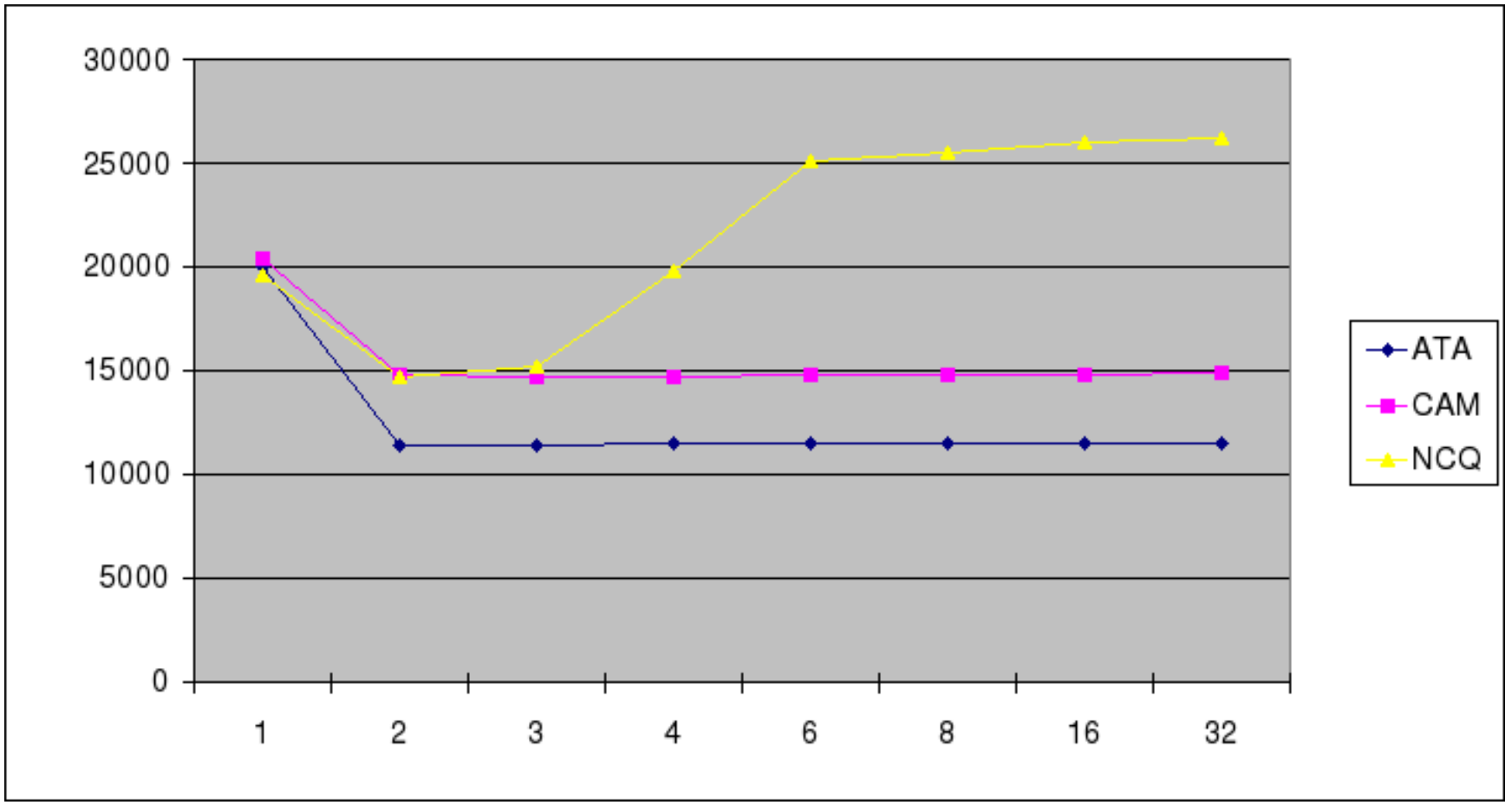
- How it looks now (camcontrol identify):

```
%camcontrol identify ada1
pass4: <OCZ-VERTEX 1.30> ATA/ATAPI-7 SATA 2.x device

protocol              ATA/ATAPI-7 SATA 2.x
device model          OCZ-VERTEX
serial number         H262LML036XYSDZVG1JI
firmware revision     1.30
cylinders             16383
heads                 16
sectors/track         63
LBA supported         125045424 sectors
LBA48 supported       125045424 sectors
PIO supported         PIO4
DMA supported         WDMA2 UDMA6
overlap not supported

Feature                      Support  Enable    Value           Vendor
write cache                    yes      yes
read ahead                     yes      yes
Native Command Queuing (NCQ)  yes              31/0x1F
Tagged Command Queuing (TCQ)  no       no       31/0x1F
SMART                          yes      yes
microcode download             yes      yes
security                       yes      no
power management               yes      yes
advanced power management      no       no       0/0x00
automatic acoustic management  no       no       0/0x00  0/0x00
```

- Performance:
  - Number of linear 512b reads per second from OCZ Vertex SSD on ICH8 AHCI HBA, for different number of threads. Generic ATA(4), CAM w/o NCQ and CAM with NCQ.



- Burst performance:
  - 2xSiI3124 PCI-X cards with 10 drives on 4 PMPs and give up to 1GB/s burst performance. Generic ata(4) gave about 240MB/s.

- Questions?