Serial ATA (SATA) Interface

Jinkyu Jeong (jinkyu@skku.edu)
Computer Systems Laboratory
Sungkyunkwan University
http://csl.skku.edu
Solid State Drive (SSD)
# Commercial SSDs

<table>
<thead>
<tr>
<th>Rank</th>
<th>Model</th>
<th>Specifications</th>
<th>Price (as of Feb. 4, 2017)</th>
<th>Discount</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Samsung 750 EVO</td>
<td>16nm TLC NAND, MGX controller, 16nm</td>
<td>$206,630</td>
<td>36%</td>
</tr>
<tr>
<td>2</td>
<td>Samsung 850 PRO</td>
<td>3D V-NAND, 40nm, 3 years A/S</td>
<td>$1,163,970</td>
<td>25%</td>
</tr>
<tr>
<td>3</td>
<td>SanDisk SSD PLUS</td>
<td>SSD PLUS, 15 years warranty</td>
<td>$311,550</td>
<td>39%</td>
</tr>
</tbody>
</table>

*Disclaimer: The prices are subject to change and may vary. Please visit the official website for the latest information.*

Anatomy of an SSD

- Samsung 850 Evo

[Diagram showing the components of an SSD: NAND Flash, SSD Controller, DRAM]

HDDs vs. SSDs

<table>
<thead>
<tr>
<th>Category</th>
<th>840 SSD (500GB)</th>
<th>Category</th>
<th>2.5” SATA HDD (500GB, 7200rpm)</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAND FLASH</td>
<td>540 / 330</td>
<td>seq. R/W Speed (MB/s)</td>
<td>60 / 160 (*140 / 70)</td>
<td>x3<del>8 / 2</del>5</td>
</tr>
<tr>
<td>98,000 / 70,000</td>
<td>0.1</td>
<td>ran R/W Speed (IOPS)</td>
<td>450 / 400</td>
<td>217 / 175</td>
</tr>
<tr>
<td>Data Access Time (ms)</td>
<td>78,700</td>
<td>data access time (ms)</td>
<td>10~12</td>
<td>x100~120</td>
</tr>
<tr>
<td>Benchmark Score (PCMark Vantage)</td>
<td>0.127W (active)</td>
<td>power consumption (operation)</td>
<td>1.75W</td>
<td>x13 ↓</td>
</tr>
<tr>
<td>Power Consumption (Operation)</td>
<td>0.046W (Idle)</td>
<td>idle power</td>
<td>0.8W</td>
<td>x17 ↓</td>
</tr>
<tr>
<td>Vibration</td>
<td>20G (10~2000Hz)</td>
<td>vibration</td>
<td>0.5G (22~350Hz)</td>
<td>x40</td>
</tr>
<tr>
<td>Shock (Operation)</td>
<td>1500G/0.5ms</td>
<td>shock</td>
<td>350G/2.0ms</td>
<td>x4</td>
</tr>
<tr>
<td>Reliability (MTBF*)</td>
<td>1.5M hours</td>
<td>reliability (MTBF*)</td>
<td>700k hours</td>
<td>x2</td>
</tr>
</tbody>
</table>

State of the Art

• World’s first 2.5” SAS 32TB SSD @ Flash Memory Summit 2016

Source: THESSDREVIEW, Samsung Newsroom
(Messy) Storage Interfaces

J. Pappas, Annual Update on Interfaces, FMS, 2015.
Moving Closer to the Processor

ST-506 Interface

- ST-506: The first 5.25” HDD from Seagate
- 1980, up to 5MB at $1500
(Parallel) ATA Interface

- IDE (Integrated Disc Electronics) drives
- Support for two drives (master/slave)
- Standardized in 1994 as ANSI X3.221: AT Attachment Interface for Disk Drives
Serial ATA (SATA)

• Primary internal storage interconnect for desktop and mobile PCs
  – HDDs, SSDs, optical drives, removable disks, ..

• More than 1.1 billion SATA drives shipped during 2001-2008

• Market share (as of 2008):
  – Desktop PC market: 99%
  – Mobile PC market: 97.7%
  – Enterprise market: 27.6%

• Serial, point-to-point, half duplex
Why SATA?

• Lower pin count (cost, space, …)
• Lower voltage support (5V $\rightarrow$ 0.7V)
• Higher performance:
  – SATA 1: 150MB/s @ 1.5Gb/s
  – SATA 2: 300MB/s @ 3Gb/s
  – SATA 3: 600MB/s @ 6Gb/s
• Simple drive configuration (no slave)
• Greater reliability (CRC/packet)
• Migration to servers (hot plug, NCQ, …)
SCSI

• Small Computer System Interface
  – SASI (Shugart Associates System Interface), 1981
  – SASI was adopted as a standard and named SCSI in 1986

• A set of standards for physically transferring data between computers and peripherals
  – Parallel, shared bus
  – Hard disk drives, tape drives, scanners, CD drivers, ...

• SCSI evolution
  – SCSI-1: 8-bit, 5MB/s, 1986
  – SCSI-2: 16-bit, 10MB/s, 1989
  – SCSI-3: 16-bit, up to 320MB/s (Ultra-320), 1992-2001
SAS

• **Serial Attached SCSI**
  – Serial, point-to-point, full-duplex interconnect for SCSI
  – Backward compatible with SATA 2 and later: SATA drives can be connected to SAS backplanes

• **SAS evolution**
  – SAS-1: 3Gbps, 2005
  – SAS-3: 12Gbps, 2013

• **Being used for server storage applications**
  – Performance, reliability, sophisticated software control, and extensive error reporting
NVMe (NVM Express)

• The industry standard interface for high-performance NVM storage
  – NVMHCI 1.0 in 2008 (led by Intel)
  – NVMe 1.0 in 2011 by NVM Express Workgroup
  – NVMe 1.2 in 2014
  – Supported by major OSes: Windows, Linux, Solaris, ...

• PCIe-based

• Commercial NVMe SSDs
  – Samsung XS1715, Intel DC P3700, HGST SN150, …
  – ~ 75% (Client) or 37% (Enterprise) of SSD shipments by 2018
SSD Form Factors

• Started with non-HDD form factors

• Mass adoption with HDD form factors
SSD Form Factors: SATA

- Proliferating in different form factors

1. Slim SATA MO-297
2. mSATA Mini MO-300B
3. mSATA MO-300A
4. MO-276 µSSD
5. 2242
6. 2260
7. 2280
8. M.2 (SATA)
SSD Form Factors: PCIe

Add-in-card (AIC)

- M.2 (PCIe: Up to x4)
  - 2242
  - 2260
  - 2280

- U.2 (SFF-8639: Up to x4)

- SATA
- SAS
Serial ATA (SATA) Interface
PATA Block Diagram
PATA Programming Interface

• Task file
  – A block of registers mapped into the x86 IO address space
  – Eight Command Registers (@ 0x01F0) + One Control Register (@ 0x03F6)

• Issuing a command
  – Initialize the task file for a particular command
  – Write the appropriate value to the Command register @ 0x01F7
Physical Drive Geometry

- **Track**
- **Sector**
- **Heads** (2 per platter)
- **Cylinder**: All corresponding tracks on other platters
- **Spindle**
## Taskfile (Physical Addressing)

<table>
<thead>
<tr>
<th>Cmd Reg</th>
<th>Reads</th>
<th>Writes</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Address</td>
<td>7</td>
<td>0 7</td>
<td>16-bit accesses</td>
</tr>
<tr>
<td>01F0</td>
<td>Data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>01F1</td>
<td>Error</td>
<td>Feature</td>
<td>8-bit access only</td>
</tr>
<tr>
<td>01F2</td>
<td></td>
<td>Sector Count</td>
<td>8-bit access only</td>
</tr>
<tr>
<td>01F3</td>
<td></td>
<td>Sector Number</td>
<td>8-bit access only</td>
</tr>
<tr>
<td>01F4</td>
<td></td>
<td>Cylinder Low</td>
<td>8-bit access only</td>
</tr>
<tr>
<td>01F5</td>
<td></td>
<td>Cylinder High</td>
<td>8-bit access only</td>
</tr>
<tr>
<td>01F6</td>
<td>Device</td>
<td>Head</td>
<td>8-bit access only</td>
</tr>
<tr>
<td>01F7</td>
<td></td>
<td>Status</td>
<td>Command</td>
</tr>
<tr>
<td>Ctrl Reg</td>
<td>Alternate Status</td>
<td>Device Control</td>
<td>8-bit access only</td>
</tr>
</tbody>
</table>

ICE3028: Embedded Systems Design, Fall 2018, Jinkyu Jeong (jinkyu@skku.edu)
# Taskfile (28-bit LBA)

<table>
<thead>
<tr>
<th></th>
<th><strong>Cmd Reg</strong></th>
<th><strong>Reads</strong></th>
<th><strong>Writes</strong></th>
<th><strong>Notes</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Address</strong></td>
<td>7</td>
<td>0</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>01F0</td>
<td></td>
<td>Data</td>
<td></td>
<td>16-bit accesses</td>
</tr>
<tr>
<td>01F1</td>
<td>Error</td>
<td>Feature</td>
<td></td>
<td>8-bit access only</td>
</tr>
<tr>
<td>01F2</td>
<td></td>
<td></td>
<td>Sector Count</td>
<td>8-bit access only</td>
</tr>
<tr>
<td>01F3</td>
<td></td>
<td>LBA Low (7:0)</td>
<td></td>
<td>8-bit access only</td>
</tr>
<tr>
<td>01F4</td>
<td></td>
<td>LBA Middle (15:8)</td>
<td></td>
<td>8-bit access only</td>
</tr>
<tr>
<td>01F5</td>
<td></td>
<td>LBA High (23:16)</td>
<td></td>
<td>8-bit access only</td>
</tr>
<tr>
<td>01F6</td>
<td>Device</td>
<td>LBA (27:24)</td>
<td>Device LBA (27:24)</td>
<td>8-bit access only</td>
</tr>
<tr>
<td>01F7</td>
<td>Status</td>
<td>Command</td>
<td></td>
<td>8-bit access only</td>
</tr>
<tr>
<td><strong>Ctrl Reg</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>03F6</td>
<td>Alternate Status</td>
<td>Device Control</td>
<td></td>
<td>8-bit access only</td>
</tr>
</tbody>
</table>
48-bit LBA

- 28-bit LBA: up to 128GB
- ATA-6 introduced 48-bit LBA: up to 128PB
  - Two writes issued to LBA low/middle/high (0x01F3-0x01F5) and sector count (0x01F2)
  - High-order byte is loaded first
  - cf. ReadDMAExtended, WriteDMAExtended

<table>
<thead>
<tr>
<th>01F1</th>
<th>Error</th>
<th>Feature</th>
<th>Two 8-bit accesses</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Sector Count</td>
<td>Two 8-bit accesses</td>
</tr>
<tr>
<td>01F2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01F3</td>
<td></td>
<td>LBA Low (31:24 then 7:0)</td>
<td>Two 8-bit accesses</td>
</tr>
<tr>
<td>01F4</td>
<td></td>
<td>LBA Middle (39:32 then 15:8)</td>
<td>Two 8-bit accesses</td>
</tr>
<tr>
<td>01F5</td>
<td></td>
<td>LBA High (47:40 then 23:16)</td>
<td>Two 8-bit accesses</td>
</tr>
<tr>
<td>01F6</td>
<td>Device</td>
<td>Device</td>
<td>8-bit access only</td>
</tr>
</tbody>
</table>
PATA Command Sequence

1. Host Software Initializes File Registers
2. Device Updates File Registers
3. Host Software Writes to CMD Register
4. Device Parses Command
4. Device Parses Command
6. Interrupts for Data Transfer
5. Device Processes Command and Transfers Data
7. Device Updates Task File and Asserts Interrupt
8. Parse Interrupt
9. Read Status Reg
SATA Compatibility with PATA

- HBA sends the register contents to the drive via a packet called a “Frame Information Structure (FIS)”
SATA Block Diagram

<table>
<thead>
<tr>
<th>SATA 1</th>
<th>SATA 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCR[0]</td>
<td>Status register</td>
</tr>
<tr>
<td>SCR[1]</td>
<td>SError register</td>
</tr>
<tr>
<td>SCR[2]</td>
<td>SControl register</td>
</tr>
<tr>
<td>SCR[3]</td>
<td>SActive register</td>
</tr>
<tr>
<td>SCR[4]</td>
<td>SNotification register</td>
</tr>
<tr>
<td>SCR[5]</td>
<td>Reserved</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>SCR[15]</td>
<td>Reserved</td>
</tr>
</tbody>
</table>
FIS Types

• Each FIS has an 8-bit ID (type)
• Size is always multiples of 4 bytes (=1 DW)

<table>
<thead>
<tr>
<th>FIS Type</th>
<th>ID</th>
<th>Direction</th>
<th>Size</th>
<th>Generation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register FIS</td>
<td>27h</td>
<td>HBA to Device</td>
<td>5 DWs</td>
<td>Gen 1</td>
</tr>
<tr>
<td>Register FIS</td>
<td>34h</td>
<td>Device to HBA</td>
<td>5 DWs</td>
<td>Gen 1</td>
</tr>
<tr>
<td>Set Device Bits with Active field</td>
<td>A1h</td>
<td>Device to HBA</td>
<td>2 DWs</td>
<td>Gen 1, Gen 2, Gen 2</td>
</tr>
<tr>
<td>with Event Notification field</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PIO Setup</td>
<td>5Fh</td>
<td>Device to HBA</td>
<td>5 DWs</td>
<td>Gen 1</td>
</tr>
<tr>
<td>DMA Activate</td>
<td>39h</td>
<td>Device to HBA</td>
<td>1 DWs</td>
<td>Gen 1</td>
</tr>
<tr>
<td>First Party DMA Setup with Auto Activation</td>
<td>41h</td>
<td>Bidirectional</td>
<td>7 DWs</td>
<td>Gen 1, Gen 2</td>
</tr>
<tr>
<td>Data</td>
<td>46h</td>
<td>Bidirectional</td>
<td>2049 DWs</td>
<td>Gen 1</td>
</tr>
<tr>
<td>BIST Activate</td>
<td>58h</td>
<td>Bidirectional</td>
<td>3 DWs</td>
<td>Gen 1</td>
</tr>
</tbody>
</table>
# Register FIS H2D

<table>
<thead>
<tr>
<th>DW 0</th>
<th>DW 1</th>
<th>DW 2</th>
<th>DW 3</th>
<th>DW 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>+3</td>
<td>+2</td>
<td>+1</td>
<td>+0</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Features</td>
<td>Command</td>
<td>CR</td>
<td>R</td>
<td>Reserved</td>
</tr>
<tr>
<td>Device</td>
<td>LBA High</td>
<td>LBA Middle</td>
<td>LBA Low</td>
<td></td>
</tr>
<tr>
<td>Features (exp)</td>
<td>LBA High (exp)</td>
<td>LBA Mid (exp)</td>
<td>LBA Low (exp)</td>
<td></td>
</tr>
<tr>
<td>Control</td>
<td>Reserved (0)</td>
<td>Sec Count (exp)</td>
<td>Sector Count</td>
<td></td>
</tr>
<tr>
<td>Reserved (0)</td>
<td>Reserved (0)</td>
<td>Reserved (0)</td>
<td>Reserved (0)</td>
<td></td>
</tr>
</tbody>
</table>
Data FIS

Dwords of Data
(min 1 Dword max 2048 Dwords)
Non-Data Commands

Software delivers command to Shadow Registers & HBA sends Register FIS

Shadow Registers Updated & interrupt sent to system to report command completion

Host - to - device register FIS

Device - to - host register FIS

Update ATA Register

Process Command

Update task file registers & send Register FIS
DMA Read Command

1. Host initializes DMA controller.
2. Host initializes shadow task file registers (BSY=0)
3. Host writes command register (BSY=1)
4. Parse command
5. Process command
6. Send all Data Requested
7. Data received from drive
8. DMA controller transfers read data to memory
9. Update completion status within task file and report to HBA
10. Update shadow registers (DRQ=0, BSY=0)
11. Interrupt host
DMA Write Command

1. Host software initializes the DMA Controller
2. Host initializes shadow task file registers (BSY=0)
3. Writes Command register (BSY=1)
4. Parse command
5. Device notifies HBA that it’s ready to receive data
6. Host Activates DMA Controller
7. Host sends data to device
8. Data accepted by device & written to disk
9. Completion status reported and the Interrupt Pending bit is set.
10. The HBA updates shadow registers with completion status and sends interrupt
SATA NCQ

- Enqueue up to 32 commands in the drive
- Process them in an out-of-order fashion

**Native Command Queuing**
- Requested Read: A, B, C, D
- NCQ Reordered Read: B, D, A, C

**Legacy Command Non-Queued**
- Requested Read: A, B, C, D
- Non-reordered Read: A, B, C, D

**Complete**
- (1.25 revolutions)
### Enabling NCQ

- **IDENTIFY DEVICE command:**

```plaintext
<table>
<thead>
<tr>
<th>Word Offset</th>
<th>R/O</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - 74</td>
<td></td>
<td>Defined in ATA/ATAPI-7</td>
</tr>
<tr>
<td>75</td>
<td>Optional</td>
<td>Queue Depth</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4-0 Maximum queue depth-1</td>
</tr>
<tr>
<td>76</td>
<td></td>
<td>Serial ATA Capabilities</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8 Native Command Queuing supported</td>
</tr>
<tr>
<td>77</td>
<td>Reserved</td>
<td>Reserved for SATA</td>
</tr>
<tr>
<td>78</td>
<td>Optional</td>
<td>Serial ATA Features</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 DMA Setup FIS non-zero buffer offsets supported</td>
</tr>
</tbody>
</table>
```

*ICE3028: Embedded Systems Design, Fall 2018, Jinkyu Jeong (jinkyu@skku.edu)*
First Party DMA Commands

- First Party DMA (FPDMA) Read

<table>
<thead>
<tr>
<th>Register</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Features</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Features (exp)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sector Count</td>
<td>7-0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sector Count (exp)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sector Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>Sector Number (exp)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>Cylinder Low</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>Cylinder Low (exp)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>Cylinder High</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>23</td>
<td></td>
</tr>
<tr>
<td>Cylinder High (exp)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>40</td>
<td></td>
</tr>
<tr>
<td>Device/Head</td>
<td>FUA</td>
<td>1</td>
<td>Res</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>60h</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- First Party DMA (FPDMA) Write
FPDMA Read Command

1. Host initializes shadow task file registers (BSY=0), writes the command, and sets the tag bit in SActive Reg (BSY=1)

4. Host receives bus release & can send new cmd

6. Host receives notification that read data is ready & initializes DMA engine

8. Host receives data

10. Host Clears pending tag bit in SActive register

2. Parse command (60h)

3. Device releases bus by sending Register FIS (BSY=0, REL=1)

5. Device reads data from disk sends DMA Setup FIS

7. Device delivers data to the HBA

9. Device reports completion of Command by returning status and setting tag bit.
FPDMA Write Command

1. Host initializes the shadow task file registers (BSY=0), writes Cmd register, and sends the Register FIS (BSY=1)

4. Host receives release and sets the tag bit in SActive Reg

6. Host initializes DMA engine

8. Host receives notification that drive is ready for data
   **Auto-activate**

9. Host sends the Data FIS

12. Host Clears bit in SActive register & sends interrupt

2. Drive parses the command (61h)

3. Device releases the bus (BSY=0 and Rel=1)

5. Drive is ready to receive data and sends the DMA Setup FIS

7. Drive sends the DMA Activate

10. Data received and written to disc

11. Device reports completion of Command by updating status and setting tag bit

ICE3028: Embedded Systems Design, Fall 2018, Jinkyu Jeong (jinkyu@skku.edu)
AHCI

- **Advanced Host Controller Interface**
  - By Intel
  - The current version is AHCI v1.3 (Oct. 2010)
  - Defines the functional behavior and software interface of the SATA/AHCI adapters
  - AHCI HBAs support from 1 to 32 ports
  - An HBA optionally supports SATA NCQ via the FPDMA Queued Command protocol for each device of up to 32 entries
Command Queue in AHCI

Memory-Mapped Registers

- HBA Capability/Control/Status
- Reserved
- Vendor Specific

Port 0
Port 1
Port 31

Command List (Port 0)
Cmd 0
Cmd 1
... Cmd 31

Command Table
FPDMA Command (H2D Reg FIS)
ATAPI Command (H2D Reg FIS)
Reserved

Physical Region Descriptors (Scatter/Gather List)
- Data Base Addr
- Byte Count
- Maximum 65535 entries

Command List (Port 31)

Command Table