SSD Firmware Implementation Project
- Lab. #4 -

Sang-Phil Lim (lsfeel0204@gmail.com)
SKKU VLDB Lab.

2011-04-28
# Lab. Time Schedule

<table>
<thead>
<tr>
<th>Lab.</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>FTL Simulator Development Guide</td>
</tr>
<tr>
<td>#2</td>
<td>FTL Simulation Guide</td>
</tr>
<tr>
<td>#3</td>
<td>Project 1 Presentation</td>
</tr>
<tr>
<td>#4</td>
<td>Jasmine OpenSSD platform tutorial #1</td>
</tr>
<tr>
<td>#5</td>
<td>Jasmine OpenSSD platform tutorial #2</td>
</tr>
<tr>
<td>#6</td>
<td>FTL Porting Guide</td>
</tr>
<tr>
<td>#7</td>
<td>Firmware Debugging Guide</td>
</tr>
<tr>
<td>#8</td>
<td>SSD Performance Evaluation Guide</td>
</tr>
<tr>
<td>#9</td>
<td>Project 2 Presentation</td>
</tr>
</tbody>
</table>
Now, it is time to move on the "REAL Environment"
The OpenSSD Project

• It is an initiative to promote research and education on the recent SSD (Solid State Drive) technology by providing easy access to OpenSSD platforms on which open source SSD firmware can be developed
  – http://www.openssd-project.org/
INDILINX
Jasmine Platform
- H/W perspective
Jasmine Reference Board

- Flash Memory Socket (DIMM) (front #4, rear #4)
- UART port
- JTAG debug port
- 64MB SDRAM
- SATA 2.0 (3Gbps)
- Indilinx Barefoot™ SSD Controller
- NAND Flash Module (K9LCG08U1M, Samsung)

Embedded Systems Design: Lab #4
Jasmine Reference Board

**Embedded Systems Design: Lab #4**

**Diagram: Jasmine Reference Board**

- **INDILINX Barefoot™ Controller**
  - SRAM (96KB) Controller
  - ROM Controller
  - ARM7TDMI-S Core
  - Clock Generator
  - APB Bridge

- **AHB**
  - NAND Controller
  - Buffer Manager
  - SATA Device
  - DRAM Controller
  - Memory Utility

- **DRAM Access Bus**
  - NAND Flash
  - SATA Host interface
  - DRAM
  - JTAG debug port

- **Cores and Controllers**
  - UART
  - GPIO
  - Timer
  - WDT
  - PMU
  - ICU
  - JTAG
1. Memory Map

• Overview

![Memory Map Diagram]
1. Memory Map

• DRAM footprint
2. SATA Controller

- Handling IO request from host
  - By FIQ handler
- SATA NCQ (Native Command Queuing)
  - Queuing incomplete command
  - 32 queue depth, FIFO
- SATA command queue
  - Queuing complete command
  - 128 entry, FIFO
2. SATA Controller

- SATA NCQ & SATA command queue
3. Buffer Manager

- Buffer frame allocation & management
4. Memory Utility & ECC Engine

• DRAM access limitation
  – E.g. `mem_copy`
5. NAND Flash Architecture

- Multi-ch/Multi-way architecture
  - 4 channel
  - 8 way
5. NAND Flash Architecture

- **Bank**
  - 16 Bit IO bus
  - High/Low NAND chips
- **Virtual block/page**
  - In 2-plane mode,
    - Physical page size x 4
    - Physical block size x 4
5. NAND Flash Architecture

- Interleaved IO (2-plane mode)
  - 33 44 55 66 ........ 77 88 99 AA ........

![Diagram of NAND Flash Architecture]
Ref) Two-plane Page Program

• Timing diagram

**Figure 15. Two-Plane Page Program**

- **R/B**
- **I/Oe – 7**
- **60h** Address & Data Input
- **11h**
- **81h** Address & Data Input
- **10h**
- **70h**

**NOTE**: 1. It is noticeable that physically same row address is applied to two planes. 2. Any command between 11h and 81h is prohibited except 70hF1hF2 and FFh.

**Data Input**

- Plane 0 (2048 Block)
  - Block 0
  - Block 1
  - Block 2

- Plane 1 (2048 Block)
  - Block 1
  - Block 2

---

Embedded Systems Design: Lab #4
5. NAND Flash Architecture

• Page copy-back
  – FC_CPBACK
  – FC_MODIFY_CPBACK
6. NAND Flash Controller

• Handling flash command
6. NAND Flash Controller

- **FCP**
  - For setting flash cmd.
- **BSP**
  - For debugging
Getting Started to Develop an FTL
1. Development Environment Setup

• Hardware Requirement

• Software Requirement
  – Jasmine Firmware Software (latest release)
    • http://www.openssd-project.org/wiki/Downloads
  – MS Visual Studio Express Free Edition 2010
  – Code Sourcery G++ Lite Edition for ARM
1. Development Environment Setup

• Hardware Setting
1. Development Environment Setup

• Software Setting
  – Install compilation tool
    • MS Visual Studio Express Free Edition 2010
      – To build the firmware installer (install.exe)
    • Code Sourcery G++ Lite Edition for ARM
      – To build Firmware binary file (firmware.bin)
  – Setting for serial communication
    • Hyper terminal (BAUD_115200/8/N/1/X)
2. Compile & Build firmware

• Build the firmware software using GNU tools

  > cd ./build_gnu
  > build.bat

• Compile the installer
  – Open ./installer/installer.sln & Build
  – Move ./installer/install.exe to ./build_gnu
3. Install Firmware to Jasmine Board

- Booting the Jasmine board as ‘Factory mode’

- Install Firmware

```
> ./build_gnu/install.exe
```
3. Install Firmware to Jasmine Board

- Install for the first time
  1 – 2 – 6 – 3

- Reinstall
  1 – 2 – 3
4. Run Firmware

• Booting the Jasmine board as ‘Normal mode’
  – Unplug SATA cable
    • Jasmine would be busy doing internal low-level format
  – Plug SATA cable when LED at D4 position is lit
• Now that Jasmine is ready to process SATA command
  • Try to send IO requests to Jasmine board! 😊
Jasmine Technical Document

• You can download on the OpenSSD Project homepage
  – [http://www.openssd-project.org](http://www.openssd-project.org)

• Technical Reference Manual
  – Jasmine Board Specification
  – INDILINX Barefoot™ SSD Controller Specification
  – Jasmine Firmware Architecture
  – Jasmine Firmware Software Specification

• FTL Developer’s Guide
  – FTL Porting Guide
  – Compile, Build & Install Firmware
  – Debugging Tips
Any Questions?