

CSGE602055 Operating Systems

CSF2600505 Sistem Operasi

Week 03: I/O, BIOS, Loader, & Systemd

Rahmat M. Samik-Ibrahim

University of Indonesia

<http://rms46.vlsm.org/2/207.html>

Always check for the latest revision!

REV119 20-FEB-2018

Operating Systems 2018-1 (Room 3114 Tue/Thu)

Class: A (10:00-12:00) | B (13:00-15:00) | C (16:00-18:00)

Week	Schedule	Topic	OSC9
Week 00	06 Feb - 12 Feb 2018	Intro & Review1	Ch. 1, 16
Week 01	13 Feb - 19 Feb 2018	Review2 & Scripting	Ch. 1, 2
Week 02	20 Feb - 26 Feb 2018	Protection, Security, Privacy, & C-language	Ch. 14, 15
Week 03	27 Feb - 05 Mar 2018	I/O, BIOS, Loader, & Systemd	Ch. 13
Week 04	06 Mar - 12 Mar 2018	Addressing, Shared Lib, & Pointer	Ch. 8
Week 05	13 Mar - 19 Mar 2018	Virtual Memory	Ch. 9
Reserved	20 Mar - 24 Mar 2018		
Mid-Term	26 Mar - 03 Apr 2018	(UTS)	
Week 06	05 Apr - 11 Apr 2018	Concurrency: Processes & Threads	Ch. 3, 4
Week 07	12 Apr - 18 Apr 2018	Synchronization	Ch. 5, 7
Week 08	19 Apr - 25 Apr 2018	Scheduling	Ch. 6
Week 09	26 Apr - 05 May 2018	File System & Persistent Storage	Ch. 10, 11, 12
Week 10	07 May - 16 May 2018	I/O Programming & Network Sockets Programming	
Reserved	17 May - 22 May 2018		
Final	23 May - 26 May 2018	(UAS)	
Deadline	07 Jun 2018 16:00	Extra assignment deadline	

● Operating Systems Check List

- ☐ Trace this document from <http://rms46.vlsm.org/2/207.html>
- ☐ Have a decent OS Book and map it to **OSC9**.
- ☐ Create **public** project "os181" on your github.com account.
 - ☐ Write in "README.md" file:
 - Special for Week 00: "**ZCZC Sistem Operasi 2018 Awal (1)**".
 - Add line on Week01: "**ZCZC W01 ...**".
 - On Week02: "**ZCZC W02 ...**".
 - On WeekXX: "**ZCZC WXX ...**".
- ☐ Encode your **QRC** with image size of approximately 250x250 pixels:
"**OS181 CLASS ID GITHUB-ACCOUNT SSO-ACCOUNT SIAK-Full-Name**"
Special for Week 00: Mail your **embedded** QRC to: os181@vlsm.org
with Subject: [W00] CLASS ID SIAK-NAME.
- ☐ Write your Memo (with QRC) **every week**.
- ☐ Using your **SSO** account, login to badak.cs.ui.ac.id via kawung.cs.ui.ac.id.
 - ☐ Check folder badak:///extra/Week00/
 - ☐ Week00: Copy the folder to your home directory:

```
cp -r /extra/Week00/W00-demos/ W00-demos/
```
 - ☐ For WeekXX: Copy the folder to your home directory:

```
cp -r /extra/WeekXX/WXX-demos/ WXX-demos/ (XX=01, 02,... 10).
```
- ☐ **How to improve this document?**

Agenda

- 1 Start
- 2 Agenda
- 3 Week 03
- 4 Legacy BIOS
- 5 UEFI
- 6 UEFI Boot
- 7 Operating System (Boot) Loader
- 8 GRUB Map
- 9 init (SYSV legacy)
- 10 UpStart - Ubuntu
- 11 The All New "systemd"
- 12 systemctl
- 13 PCH: Platform Controller Hub
- 14 Some Terms
- 15 I/O
- 16 The End

Week 03: I/O, BIOS, Boot, & Systemd

- Reference: (OSC9-ch13 demo-w03)
- Firmware
 - BIOS: Basic Input Output System.
 - UEFI: Unified Extensible Firmware Interface.
 - ACPI: Advanced Configuration and Power Interface.
- Operating System (Boot) Loader
 - BOOTMGT: Windows Bootmanager / Bootloader.
 - LILO: Linux Loader.
 - GRUB: GRand Unified Bootloader.
- Operating System Initialization
 - Init (legacy)
 - UpStart
 - Systemd
- I/O
 - Interrupt.
 - DMA.
 - ETC.

- Check Settings.
- Initialize CPU & RAM.
- POST: Power-On Self-Test.
- Initialize ports, LANS, etc.
- Load a Boot Loader.
- Handover to the Boot Loader.
- Provides "Native" (obsolete) Drivers only (not loadable).
- Provides "INT" services .
- Limitation.
 - Technology of 1970s.
 - 16 bits software.
 - 20 bits address space (1 MB).
 - 31 bits disk space (2 TB).

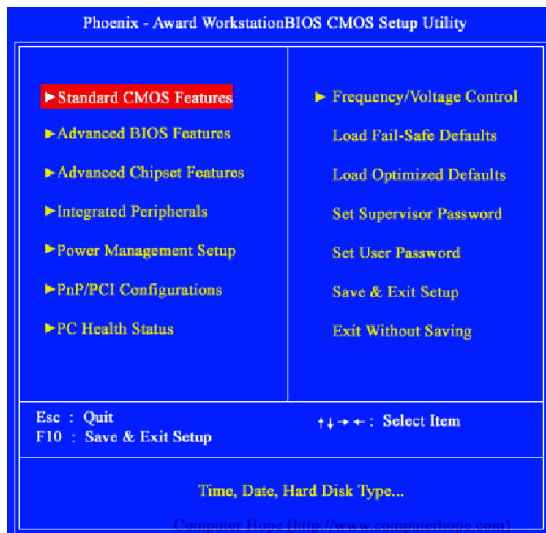


Figure: BIOS

- A Firmware Specification, not an Implementation!
- No (INT) service after boot.
- HII: Human Interface Infrastructure.
- Protected Mode.
- Flexible.
 - Technology of 2000s.
 - written in C.
 - (third party) loadable drivers and tools.
 - Emulate Legacy BIOS transition (MBR block, INT service).
 - UEFI Shell: environment shell for diagnostic (no need for DOS).
- Problems
 - Who controls the Hardware?
 - Is "Secure Boot" a good thing?
 - How about a **NASTY/LOCKING/TROJAN** UEFI implementation?
 - Different **DRIVERS**.



Figure: UEFI

Platform Initialization (PI) Boot Phases

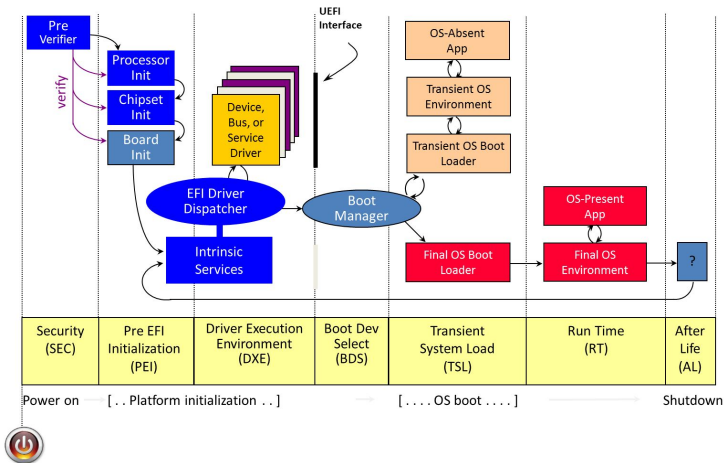


Figure: UEFI Boot Process¹.

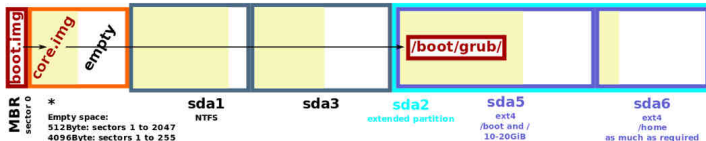
Operating System (Boot) Loader

- General
 - How/Where to start the operating system?
 - What to do?
 - How many ways to boot?
 - How many types of OS?
- GRUB/GRUB2: GRand Unified Boot system
 - Stage 1 (boot.img): MBR (Master Boot Record) – Where is everything
 - Stage 1.5 (core.img): generated from diskboot.img
 - Stage 2: Kernel Selection: Windows, Linux, BSD, etc.
- GRUB2
 - More flexible than GRUB legacy
 - More automated than GRUB legacy
- Disk Partition
 - MBR: Master Boot Record (1983).
 - GPT: GUID Partition Table (2010s).

GNU GRUB 2

Locations of *boot.img*, *core.img* and the */boot/grub* directory

Example 1: an MBR-partitioned harddisc with sector size of 512 or 4096Bytes



Example 2: a GPT-partitioned harddisc with sector size of 512 or 4096Bytes

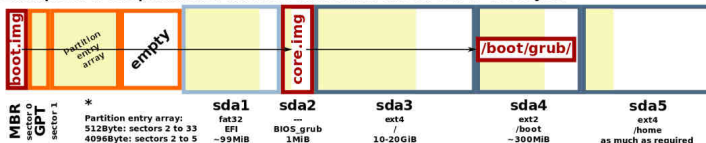


Figure: GRUB¹.

¹Source Shmuel Csaba Otto Traian 2013

init (SYSV legacy)

- File: `/etc/inittab`.
- Folders: `/etc/rcX.d` — `X` = runlevel.
 - Seven (7) different runlevels:
 - 0 (shutdown).
 - 1 (single-user/admin).
 - 2 (multi-user non net).
 - 3 (standard).
 - 4 (N/A).
 - 5 (3+GUI).
 - 6 (reboot).
 - SXX-YYY: Start
 - KXX-YYY: Kill.
- One script at a time in order.
- dependency is set manually.

- Developer: Ubuntu.
- Folder: `/etc/init/`.
- Control: `initctl`.
 - `initctl list` – listing all processes managed by upstart.
- better support for hotplug devices.
- cleaner service management.
- faster service management.
- asynchronous.

The All New "systemd"

- Replaces (SYSV) init and UpStart.
 - better concurrency handling: Faster!
 - better dependencies handling: No more "S(tarts)" and "K(ills)".
 - better crash handling: automatic restart option.
 - better security: group protection from anyone including superusers.
 - simpler config files: reliable and clean scripts.
 - hotplug: dynamic start/stop.
 - supports legacy systems (init).
 - overhead reducing.
 - unified management way for all distros.
 - bloated: doing more with more resources.
 - linux specific: NOT portable.

systemctl

```
for II in \
'systemctl list-unit-files | head -8; echo "(...)";
    systemctl list-unit-files| tail -8' \
'systemd-analyze blame | wc -l; echo "===";
    systemd-analyze blame | head -15' \
'systemctl --full | wc -l; echo "===";
    systemctl --full | head -10' \
'systemctl list-units | wc -l; echo "===";
    systemctl list-units | head -10' \
'systemctl list-units |grep .service|wc -l;echo "===";
    systemctl list-units|grep .service|head -10' \
'systemctl list-units | grep ssh.service' \
'systemctl status ssh.service' \
'systemctl is-enabled ssh' \
'journalctl' \
'journalctl -b' \
do
...
```


PCH: Platform Controller Hub

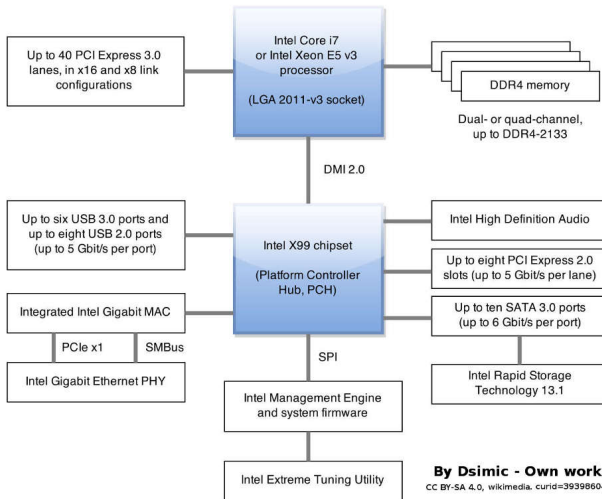


Figure: PCH: Platform Controller Hub

Some Terms

- PCH: Platform Controller Hub
- PCIe: Peripheral Component Interconnect Express — 32 bits for (16 * 1x or 8 * 2x or 4 * 4x or 2 * 8x or 1 * 16x) * (2 direction) lanes.
- DMI: Direct Media Interface. Eg. DMI 2.0 (2 GB/s; 4x)
- GT/s: GigaTransfers per second
- 1 KB (KiloByte) = 1000 bytes — 1 KiB (Kibibyte) = 1024 bytes¹
- SMB: System Management Bus
- SPI: Serial Peripheral Interface, a de facto standard bus.
- SATA: Serial AT Attachment. Eg. SATA 3.2 \approx 2 GB/s.
- DDR4 SDRAM: Double Data Rate Fourth-generation Synchronous Dynamic Random-Access Memory: 2 x DDR2 (DDR2 = 2 x DDR (DDR = 2 x SDRAM)). Eg. DDR4-3200 (8x SDRAM); Memory Clock: 400 MHz; Data Rate: 3200 MT/s; Module Name PC4-25600; Peak Transfer Rate: 25600 MB/s,

¹In IT tradition; 1 KB = 1024 bytes

I/O (1)

- Direct I/O vs. Memory Mapped I/O
- Interrupts: Non Maskable (NMI) vs Maskable (MI)
- DMA: Direct Memory Access
- I/O Structure:
 - Kernel (S/W).
 - I/O (S/W: Kernel Subsystem)
 - Driver (S/W)
 - Controller (H/W)
 - Device (H/W)
- I/O Streams
 - APP
 - HEAD
 - MODULES
 - DRIVER
 - H/W.

- I/O Interface Dimensions
 - Character-stream vs. Block;
 - Sequential vs. Random-access;
 - Sharable vs. Dedicated;
 - Parallel vs. Serial;
 - Speed;
 - Read Write – Read Only – Write Only.
 - Synchronous vs. Asynchronous;
 - Blocking vs. Non-Blocking.
- Where should a new algorithm be implemented?
 - APP?
 - Kenel?
 - Driver?
 - Controller?
 - HW?

The End

- ☐ This is the end of the presentation.
- ☒ This is the end of the presentation.
 - This is the end of the presentation.