LPD: Low Power Display Mechanism for Mobile and Wearable Devices

MyungJoo Ham (presenter),
Frontier Computer Science (FCS) Lab
Software R&D Center, Samsung Electronics

Inki Dae,
Platform Solution Lab

Chanwoo Choi
## Devices Evolving...

<table>
<thead>
<tr>
<th>Year</th>
<th>Device</th>
<th>Resolution</th>
<th>Battery Capacity</th>
<th>Processor</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010</td>
<td>Galaxy S</td>
<td>800x480</td>
<td>1500mAh</td>
<td>A8 1GHz x 1</td>
</tr>
<tr>
<td>2012</td>
<td>Galaxy S3</td>
<td>1280x720</td>
<td>2100mAh</td>
<td>A9 1.4GHz x 4</td>
</tr>
<tr>
<td>2013</td>
<td>Galaxy S4</td>
<td>1920x1080</td>
<td>2600mAh</td>
<td>K300 1.9GHz x 4</td>
</tr>
<tr>
<td>2015</td>
<td>Galaxy S6 Edge</td>
<td>2560x1440</td>
<td>2600mAh</td>
<td>A57 2.1GHz x 4 + A53 1.5GHz x 4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Year</th>
<th>Device</th>
<th>Resolution</th>
<th>Battery Capacity</th>
<th>Processor</th>
</tr>
</thead>
<tbody>
<tr>
<td>2013</td>
<td>Galaxy Gear</td>
<td>320x320</td>
<td>315mAh</td>
<td>A9 0.8GHz x 1</td>
</tr>
<tr>
<td>2014.4</td>
<td>Gear 2</td>
<td>320x320</td>
<td>300mAh</td>
<td>A7 1GHz x 2</td>
</tr>
<tr>
<td>2014.11</td>
<td>Gear S</td>
<td>480x320</td>
<td>300mAh</td>
<td>A7 1GHz x 2</td>
</tr>
</tbody>
</table>
Devices Evolving: Display Resolution

- 2010 Galaxy S
  - 800x480
  - 1500mAh
  - A8 1GHz x 1

- 2012 Galaxy S3
  - 1280x720
  - 2100mAh
  - A9 1.4GHz x 4

- 2013 Galaxy S4
  - 1920x1080
  - 2600mAh
  - K300 1.9GHz x 4

- 2015 Galaxy S6 Edge
  - 2560x1440
  - 2600mAh
  - A57 2.1GHz x 4 + A53 1.5GHz x 4

Increases memory traffic significantly. (60FPS)
- (2010) 0.09 GiB/s
- (2012) 0.21 GiB/s
- (2013) 0.46 GiB/s
- (2015) 0.82 GiB/s

...
Devices Evolving: Display Resolution

- **2010 Galaxy S**
  - Display Resolution: 800x480
  - Battery Capacity: 1500mAh
  - Processor: A8 1GHz x 1

- **2012 Galaxy S3**
  - Display Resolution: 1280x720
  - Battery Capacity: 2100mAh
  - Processor: A9 1.4GHz x 4

- **2013 Galaxy S4**
  - Display Resolution: 1920x1080
  - Battery Capacity: 2600mAh
  - Processor: K300 1.9GHz x 4

- **2015 Galaxy S6 Edge**
  - Display Resolution: 2560x1440
  - Battery Capacity: 2600mAh
  - Processor: A57 2.1GHz x 4 + A53 1.5GHz x 4

Increases memory traffic significantly. (60FPS)

- **2010** 0.09 GiB/s
- **2012** 0.21 GiB/s
- **2013** 0.46 GiB/s
- **2015** 0.82 GiB/s


In the dd (writing and reading 360MiB) test with NURI board, the memory throughput was not changed (the performance is not deteriorated) while the SoC power consumption has been reduced by 1%. When the memory access is not that intense while the CPU is heavily used, the SoC power consumption has been reduced by 6%.

With Memory-Bus DVFS... (DRAM and its interface is beginning to be important)
Devices Evolving: Display Resolution

2010
Galaxy S
800x480
1500mAh
A8 1GHz x 1

2012
Galaxy S3
1280x720
2100mAh
A9 1.4GHz x 4

2013
Galaxy S4
1920x1080
2600mAh
K300 1.9GHz x 4

2015
Galaxy S6 Edge
2560x1440
2600mAh
A57 2.1GHz x 4 +
  A53 1.5GHz x 4

Increases memory traffic significantly. (60FPS)

(2010) 0.09 GiB/s
(2012) 0.21 GiB/s
(2013) 0.46 GiB/s
(2015) 0.82 GiB/s

In the dd (writing and reading 360MiB) test with NURI board, the memory throughput was not changed (the performance is not deteriorated) while the SoC power consumption has been reduced by 1%.

When the memory access is not that intense while the CPU is heavily used, the SoC power consumption has been reduced by 6%.

With Memory-Bus DVFS... (DRAM and its interface is beginning to be important)

Applying DVFS to Memory/Bus is now effective as the traffic becomes heavy.

What if we can reduce the traffic itself?

Wearable Devices:

Small Battery (& not increasing…)
Longer Battery Life Expected (~3d)

➔ Even a small improvement is much welcomed!
Display, The Uncharted Waters

• **Energy Hogs**
  - **CPU**: DVFS, Gating, Hotplug, Power-aware Scheduler, big.LITTLE, aSMP, …
  - **GPU**: DVFS, Gating, Scheduler, …
  - **Memory**: DVFS
  - **Network** (WiFi, 3G, LTE, BT, …): Gating, Aggregating, …
  - Other peripherals (eMMC, SD, sensors, ports, PMIC, …): Gating, chip modes, …
  - **Display**: practically, not much done especially with software.

Display, The Uncharted Waters

• Many Approaches Exist
  – Work in theory, Too difficult in practice

  • Adjust Color Depth: you cannot do this w/ SW. Image quality
  • Dynamic Backlight Brightness
    – Slow Response. Image quality. Not Applicable to AMOLED
  • Dynamic Display Refresh Rate
    – Image quality (became practical recently…)
  • Compression: you cannot do this w/ SW
  • Skip Duplicated Transmission
    – Works only if the whole frame is constant

  – Or Requires Additional Hardware (ARM: ASTC, AFBC, TE, SC)
Display and Bandwidth

- Display Energy Saving = Bandwidth Reduction, Mostly.

- What if we get “LPDDR5” later?
Display and Bandwidth

• Display Energy Saving = Bandwidth Reduction, Mostly.

• What if we get “LPDDR5” later?
  – Not going to solve things…
Display and Bandwidth

- Display Energy Saving = Bandwidth Reduction, Mostly.

- What if we get “LPDDR5” later?
  - Not going to solve things...

We will waste more bandwidth with more creative ways...

http://community.arm.com/groups/arm-mali-graphics/blog/2013/10/28/saving-system-power-with-arm-multimedia-ip
To Practically Reduce Display Power,

• Tackle the Bandwidth
• Mind the overhead (it may eat up)
• (Hopefully) Don’t add hardware.
• (Hopefully) Don’t modify OS much
  – Keep the compatibility!
  – Transparent to other software components
  – Minimize the code.
Then. What LPD has done?
Hardware Architecture

- As Long As You Have i80 Display Panel (or similar), It Works.

i80 panel is used in
- Wearable: Galaxy Gear, Gear 2, Gear S, …
- Mobile: Galaxy S4, S5, S6, S6-Edge, …
- Phablet: Galaxy Note 3, Note 4, Note 4-Edge, …
Hardware Architecture

- As Long As You Have i80 Display Panel (or similar), It Works.

- Send Updated Parts Only (2, 3)!

- But. How Do We Know Which Part is Updated?
Software Architecture

• Fortunately, It’s Already Known!

APP
UI Library

X Window System

(Compositing) Window Manager

Kernel

Hardware

"XYZ is updated"
Software Architecture

- Fortunately, It’s Already Known!
- Why Don’t We Disseminate to Kernel & HW?

```
APP
UI Library
```

```
X Window System
(Compositing)
Window Manager
```

```
Kernel
Hardware
```

"XYZ is updated"
Example

Previous Frame  Next Frame  Updated Part (Data Sent for Next Frame)
Resulting Concept

- Full frame
  (320x320, 4Bpp, 30Hz)
  = ~ 12 MBps
- 8% partial update w/ LPD
  = ~ 1 MBps
Is It That Simple?

• The Idea Is Simple

• The Implementation for Practical Usage Isn’t Simple.
  – E.g., Co-work w/ Multiple Layers, Software Compatibility, and **Screen Tearing Prevention**
How Good Is It?
Power Saving, System-Wide

Table 2: Power Reduction of Real-World Applications

<table>
<thead>
<tr>
<th>App</th>
<th>LPD power saving</th>
<th>Reduced traffic</th>
</tr>
</thead>
<tbody>
<tr>
<td>W-launcher</td>
<td>0.22 mW</td>
<td>0.20%</td>
</tr>
<tr>
<td>Heart rate</td>
<td>0.39 mW</td>
<td>0.58%</td>
</tr>
<tr>
<td>Setup-wizard</td>
<td>1.14 mW</td>
<td>1.52%</td>
</tr>
<tr>
<td>Voice memo</td>
<td>2.70 mW</td>
<td>2.98%</td>
</tr>
</tbody>
</table>

7.87% of 176.47mW
Overhead, Invisible

• **Tested with Full-Frame Updates.**
  – No Benefit from LPD
  – LPD executes unconditionally

• **Power Consumption Increase Not Found**
  – 209.03 mW (no LPD) → 208.99 mW (LPD)
    • !!! Doesn’t mean that LPD saves even in this case !!!
  – No loops. No complicated API calls.
    • Just one more system call (per frame) w/ positional data already calculated by userspace middleware.
    and
    • A few simple linear calculations
No Cost. Compatible. Open Sourced

- No Additional Hardware Components.
- No Updates in Applications or Other OS Components.
  - Transparent to them
- Minimal Updates in X-Window & Kernel
  - A few lines added to X-Window & Linux Kernel (DRM Framework)
  - Upstreaming to Linux DRM Framework
  - Upstreamed Part
- Code Available at
  - [http://opensource.samsung.com](http://opensource.samsung.com)
    - Search w/ the product name of Samsung Gear series.
  - [http://review.tizen.org](http://review.tizen.org)
    - https://review.tizen.org/gerrit/gitweb?p=platform/kernel/linux-3.10.git;a=shortlog;h=refs/heads/tizen_LPD
Small Changes in Userspace: Window Mgr → X.org

- Tizen.org libdri2.git, Line 824-854 added (Step 4 of “Figure 5”)

No Additional Overhead

- No LPD

LPD
Small Changes in Userspace: X.org → Kernel

- Tizen.org xf86-video-exynos.git, Line 843-845 added (Step 5 of “Figure 5”)

  Executed once a frame.

  ➔ EASY TO MAINTAIN THE BACKWARD COMPATIBILITY!

The function has been there for years…
Changes in Kernel

• **Update DRM (Direct Rendering Manager) Framework**
  – Handle incoming partial updates
  – Coordinate device drivers

• **Update DRM Device Drivers (display panel, display controller) Accordingly**
  – Control devices (display panel, display controller)
  – Mind the timing and userspace requests
    • Ensure there will be no glitches in display due to LPD

• **Upstreaming On the Way.**
Orthogonal to Other Approaches

- LPD can be applied w/ other display power saving techniques, providing additional power saving.
  - Adjust Color Depth: simultaneously applicable
  - Dynamic Backlight Brightness: completely orthogonal
  - Dynamic Display Refresh Rate: simultaneously applicable
  - Compression: simultaneously applicable
  - Skip Duplicated Transmission: simultaneously applicable
  - ARM’s Solutions for their Video/Display/Graphics processors
    - ASTC: completely orthogonal
    - AFBC: simultaneously applicable
    - TE: completely orthogonal
    - SC: completely orthogonal
What We Have Learned.

• We May Still Have Optimization Points requiring Simple Ideas

• Cross-Layer Optimization Rocks!
  – “They may have a silver bullet they cannot use, but we can.”

• We Need to Educate App Developers!
  – With LPD,
    App drawing 10x10 every frame
    vs
    App drawing 300x300 every frame
    or
    App drawing 10x10 every frame but overwrites the whole frame with bitmap.

    Power efficiency varies according to how app is written.

  – We may look even at the UI/UX issues for power management.
    • Kernel/BSP developers: “Do we even need to talk with them??”