Finished and upcoming projects in LMG
Agenda

- Finished and Upcoming projects in LMG - Bernhard Rosenkranzer
- HiKey in AOSP Update - John Stultz
Projects finished between BKK16 and LAS16  1/2

- Memory allocator analysis
- File System analysis
- LCR updates: Switching to AOSP N and more
- gcc 6.2 support in AOSP M
- Updated clang toolchains for building AOSP N
- Increased upstream participation
- SystemUI carrier status display upstreaming
- Low-RAM LCR build to keep low-RAM patches working
- Clang-built kernel to build, boot and run Android
- OP-TEE integration in M-LCRs
- Contributing to chromium.org to: build 64bit browser for Android with clang for ARMv8, fix simd access for 64bit ARM neon clang compilation, update libjpeg_turbo to 1.4.90, update Zlib to version 1.2.8
Projects finished between BKK16 and LAS16 2/2

- ART instruction scheduling and improved instruction selection
- ART JIT investigation
- ART 32-bit performance investigation
- Hikey enablement on AOSP
- Open Graphics stack for Android enablement
- Continual rebase of AOSP patches on TIP kernel
- Continued nexus device enablement with mainline
- Implemented GBM/miniGBM based gralloc
- Upstreamed dmabuf support in Mesa Android EGL
Memory allocator analysis

- Primary focus: Reduce memory usage on low-memory devices
- Malloc implementations investigated: jemalloc, dlmalloc, nedmalloc, tcmalloc, musl malloc, TLSF, lockless allocator
- Memory analysis briefing
  - https://docs.google.com/document/d/15ycUEuplwZs0LPXMCy6yKRaNfY8QE2mA7kkEoa_o/edit#heading=h.z9n368sk0eai
- Challenges
  - Porting of atomic routines for ARM 64-bit platform
  - name mangling issues
  - C99 warnings
  - Wrapper/dummy calls for bionic integration (e.g. malloc_usable_size, malloc_disable, mallinfo etc.)
  - Other runtime issues
  - Benchmark porting - (tlsf-test, t-test)
  - Fragmentation analysis script
Memory allocator analysis

Summary

- `tcmalloc`, `jemalloc` wins for multi-threaded apps and run time performance (good amount of small pages available at runtime)
- Static size reduction for libc is improved with nedmalloc and tlsf
- `jemalloc-svelte` does not have good stand compare to `jemalloc` & `tcmalloc`
- Support issue with nedmalloc - no more support.
- Lockless allocator - under private license

Note: Rank graph is generated based on relative performance. For real numbers kindly refer to memory analysis document.
File System analysis

- Filesystems investigated: ext4, btrfs, f2fs, nilfs, squashfs
- Variants: encryption enabled/disabled, compression off/zlib/lz4
- File system analysis briefing (ongoing changes)
  - https://docs.google.com/a/linaro.org/document/d/1jam-PlV9iefnOqujzYWZoY8U9d9GnmPwda3MItxsPsU/edit?usp=sharing

- Challenges
  - Fixed build support for f2fs image generation (core.mk & image size alignment to 4096)
  - Fixed sparse raw image generation issue
    - Need to use for btrfs and nilfs
  - Image generation for btrfs, nilfs, squashfs etc..
  - Benchmark porting - bonnie, iozone
  - Partition overload scripts and long run impact scripts
File System analysis - Summary

● All relative rank graphs is available at
  ○ [https://docs.google.com/a/linaro.org/spreadsheets/d/1ctknBBVWUjrIZwS8OQcb5L8gCdCLuzktJgx-K_CMgt0/edit?usp=sharing](https://docs.google.com/a/linaro.org/spreadsheets/d/1ctknBBVWUjrIZwS8OQcb5L8gCdCLuzktJgx-K_CMgt0/edit?usp=sharing)

● F2FS/Ext4 Wins for
  ○ Small File Access (4K-1MB) + DB Access with disk data integrity
  ○ Potential use case: Industrial monitoring system, Consumer Phone, Health monitoring system

● NilFS outperforms for SQLite operations
  ○ Only cache here is, metadata/data gets updated later once get written to log file (kind of extended version of fdatasync over fsync)
  ○ Can be useful for power backed system and continuous log recording of small data (upto 4K) with good amount of storage
  ○ It quickly fill up the space if GC is not called in between. On 5GB space, it just went out of space for 1000 runs of iozone test. Do not recommended for “Embedded System”

● SquashFS : Good buffered I/O read
  ○ Can be used for read only partitions (system libraries and ro database)
File System analysis - Summary

● **BTRFS**: Large file + large RAM
  ○ **LZO** - Outperforms for block write/read operations (> 4MB)
  ○ Potential use case:
    ■ In flight entertainment system (mostly for movies/songs/images etc.)
    ■ Portable streaming & recording devices (should be power backed up)
  ○ Low lights:
    ■ High cpu utilization (more no# of threads)
    ■ Not recommended where small i/o transaction with sync is expected
    ■ Risk on power failure recovery

● Hybrid use of different file systems on multiple partitions can improve overall performance e.g.
  ○ Large read/write (movies, extra download) on BTRFS partition
  ○ All small read/write (docs, images) on f2fs/ext4 partition
  ○ All database access insert/update/delete on f2fs/nvfs partition

● Note: Yet to perform impact on file system as it ages
LCR updates

- Monthly Marshmallow based releases for HiKey and Juno
  - Always released on time and functional
  - Continuously improved automated validation
  - With help from B&B team, migrated to ci.linaro.org
  - More features added into release, like change logs, patch details, mirror support for build scripts
- OP-TEE integration in M-LCR together with fully automated validation
- Switch to EAS Kernel for juno builds
- Updated to android-7.0.0_r6 tag while transitioning to Android Nougat
- Rebased existing M-LCR patches/optimizations from Marshmallow to Nougat
- New hardware support: X15
gcc 6.2 support in AOSP M

- AOSP M can now be built with gcc 6.2 and binutils 2.27
- So far 2 out of 5 patches accepted upstream: Not a priority because AOSP will never officially support gcc > 4.9 (N and on: Only clang supported for compiling userland), but willing to take patches that don’t break things (especially for problems that might also show up in clang 4.1)
Updated clang toolchains for AOSP N

- We can build AOSP N/master with clang versions based on upstream master
- Work in progress on creating CI builds to see performance impacts
- Continuously adapting userspace to make sure it remains compatible (e.g. new warnings being added while most of AOSP is built with -Werror and new assertion failures detected) - report bugs or upstream fixes when possible
Increased upstream participation

- Every member of the AOSP Engineering team is tasked to do 10+ upstream reviews every month
- Continuing to update projects in external/ to newer upstream versions
- So far, limited acceptance upstream - problem getting enough time scheduled because of assignees being pulled off LMG tasks
- Upstream work getting more and more important:
  - Android One-style devices with a fixed /system partition becoming more common
  - Monthly security updates (and the heat some vendors are taking over not applying them in time) force people to stick closer to relevant upstream repositories
SystemUI carrier status display upstreaming

- SystemUI patch offering more options for carrier status display created by CodeAurora/Qualcomm:
  - Rebased to AOSP master
  - Cleaned up minor issues (unnecessary API changes)
  - Rebased again after AOSP N changes got merged

- Submitted upstream, not yet accepted
ION

- **Goal:** move Ion as a self-contained framework into drivers/android
  - Most likely userspace APIs at first

- **Ongoing work**
  - Clean up ioctls - Almost done
  - Platform/devicetree - Done, add more platforms
  - Cache coherency - Two steps forward, 1.5 steps back
HiKey in AOSP Update

John Stultz
<john.stultz@linaro.org>
HiKey boards

Google supports HiKey, a certified 96Board, as an Android reference board. AOSP provides kernel source and board support for HiKey to enable developers to easily create and debug new and existing peripheral drivers, do kernel development, and perform other tasks with fewer OEM encumbrances.

HiKey boards are available in 1GB RAM and 2GB RAM configurations from Lenovo:

![HiKey board](image)

**Figure 1.** HiKey board by Lenovo

Additional resources:
- HiKey schematics
- HiKey User Guide
- HiKey wiki

Running Android on HiKey
Continuing Collaboration

Working closely with folks at Google.
Submitting changes directly to AOSP Gerrit.
New Features Added

Updated to Nougat
v4.4-stable based kernel
(Staying current w/ -stable)
Suspend/Resume support
Interactive cpufreq gov
powerHAL integration
HDMI & USB Audio
Display panel mezzanine

Bootloader source integration
Abootimg support in UEFI
Unique serial no setup
“fastboot boot <kernel>”
USB Tethering
Overlay_Manager
FIQ_debugger*

* FIQ bouncing from secure mode not yet implemented, so really this is an irq_debugger
Energy Aware Scheduler Integration

Working through and documenting how to integrate EAS code (core, schedtune, schedfreq gov) in with AOSP & PowerHAL

Working with other Linaro teams to benchmark and measure power and performance impact

Collaboration between Linaro, ARM and Google got EAS merged in android-4.4 and integrated into HiKey!
OP-TEE Integration
OP-TEE kernel driver from Security Group merged into HiKey AOSP kernel

OP-TEE support in bootloader to follow
Overlay Manager

How to handle and support various hardware configurations in AOSP

Driver which switches between DeviceTree overlay objects in the DeviceTree source based on boot argument

Submitted upstream to lkml by Dmitry
Factory Images

Now, installing the latest AOSP master build is easy!
https://developers.google.com/android/nexus/images-preview#hikey

# Set the J15 3-4 jumper
$ unzip <filename>
$ cd <filename>
$ ./flashall.sh /dev/ttyUSBn
# Unset J15 3-4 jumper & reboot

That’s it! Really!
Common.git AOSP Efforts
Deep review of common/android-4.4 tree
Sending reverts for obsolete features
Appended dtb support (Image-dtb)
Integrated upstream timerslack_ns support
-stable merged android-4.4.y branches
EAS forward ported from android-3.18 to android-4.4
Prep-work for next LTS android-4.9 tree
Generic Build Integration

Not AOSP, but HiKey is also used in Rob Herring’s generic build project, which supports multiple devices (and architectures) out of a single build directory using Kconfig
TODOs

- Op-tee & Trusty coexisting in AOSP
- A/B updates partitioning & switching
- Moving forward to next 4.9 LTS kernel
- Work with Mali devs to avoid custom tweaks for Hikey
- Memory reductions to help 1GB variant
- Fixing bugs as they crop up.
Why this is useful...

Following upstream kernels
- v3.10 Released June 2013
- v3.18 Released Dec 2014
- v4.4 Released Jan 2016

Latest flagship devices (Mar 2016)
Current Nexus devices (Oct 2015)
HiKey in AOSP (Apr 2016)

2 years, 4 months
1 year, 3 months
4 months
Regressions fixed in android-4.4

- Two separate xt_qtaguid bugs
- USB eth adapter regression
- USB Configfs gadget fixes
- PTP null pointer deref fix
- Missing cpuset allow_attach hooks
- Performance regression in cgroup migration
- UID routing enum collision
Regressions found and fixed upstream

- /sys/module/mmcblk path ABI change
- iptables alignment breakage
- wlcore_op_get_expected_throughput null ptr dereference
- Missing CAP_WAKE_ALARM in AOSP userspace
- fib_rules UID routing collision w/ upstream
HiKey Upstreaming Status

4.6:
- PMIC
- thermal

4.7:
- eMMC
- uSD
- USB
- Wifi
- DRM display

4.8:
- power-key
- RTC
- media-reset
- adv7533

4.9 (queued):
- K3DMA fixes
- HDMI dts
- Reboot-reason dts,
- Pstore dts
- Kconfig fix
HiKey Upstreaming TODOs (~50 patches)

DTS changes (Pstore, reboot-reason, DSI/HDMI output)

K3DMA fixes

ADV7511 HDMI codec support

HI6210 I2S driver

High-speed uSD support

USB speed autonegotiation

Upstream Bluetooth solution for AOSP

Mali driver (we can dream!)  ~15 patches
Differing focuses

**AOSP (＆ HW Vendors)**
- Maximum power efficiency
- High performance & low latency graphics
- Only has to work on this one device
- v3.10 kernels
- Shipping to consumers in 6-12mo
- Fully enabled devices
- 2M+ line vendor patchsets

**Upstream Kernel Community**
- Long term maintainability
- One binary kernel runs on all devices
- Can’t hurt functionality on other devices
- Linus HEAD, -next or bust
- Server workloads
- Maximum throughput / capacity
- X86_64 (and everything else!)
Creating an overlap with HiKey

AOSP
(\& HW Vendors)

Affordable \& available devices

aarch64

Working against Latest -LTS \& Linus HEAD

Fully enabled device

Allows for validation of latest upstream kernel using AOSP userspace

Upstream Kernel Community
Thanks!

Questions?

<john.stultz@linaro.org>
Thank You

#LAS16
For further information: www.linaro.org
LAS16 keynotes and videos on: connect.linaro.org
ART Team Update

Serban Constantinescu
Linaro ART Team
linaro-art@linaro.org
What do we work on?

Linaro ART Tip

- Improve ART for the **next Android Release** (currently Android O)
- Patches merged into the upstream **AOSP Master**

Linaro ART Stable

- Improve ART for the **current Android Release** (currently Android N)
- Patches merged into the **Linaro Optimised Repository** (LOR)
- ART Optimisations bundled together with other changes into the **Members - Linaro Confectionary Release** (M-LCR)
What do we work on?

Android N Developer Preview
- Cherry picking and bug fixes

Android N Members-LCR
- ART Performance Improvements & other goodies

Android N
- Only bug fixes and security updates
  - E.g: 7.0_r1, 7.0_r2, ...

All patches merged upstream will be part of the next Android Release (Android O)

Q2 2016

Q3 2016

AOSP Master
Meet the Team

- **ARM Assignees:**
  - Alexandre Rames
  - Anton Kirilov
  - Artem Serov
  - Julien Duraj
  - Scott Wakeling
  - Xueliang Zhong
  - Serban Constantinescu

- **Spreadtrum Assignees:**
  - Donghui Bai
  - Tim Zhang

- **MediaTek Assignees:**
  - Tatwai Chong

- You can find us @ linaro-art@linaro.org

- Thank you all for your contributions!
Work that was done since LCA16?

● **Performance:**
  ○ Various improvements for Android Master
    ■ ~ 66 patches
  ○ Various improvements for Linaro M-LCR
    ■ ~ 46 patches

● **Infrastructure:**
  ○ **ART Testing:**
    ■ ~ 60 benchmarks
  ○ **ART Reports:**
    ■ Automated monitoring of upstream and per patch improvements
  ○ **Continuous Integration:**
    ■ Considerably improved stability and reproducibility

● **Investigations:**
  ○ 32 bit Performance Investigation
  ○ JIT Investigation

1- Documents available for Linaro Members
Work that will be done by LCA17?

● **Android Master:**
  - ARM based VIXL32 ART backends
  - Focus on 32bit performance
  - Various other improvements

● **Android N LCR:**
  - Various performance improvements

● **Infrastructure:**
  - Various ARM and competitor platforms
## ART Performance Improvements

### Linaro M-LCR vs. Android M

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Mode</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmarksgame</td>
<td>32 bit</td>
<td>4.12%</td>
</tr>
<tr>
<td></td>
<td>64 bit</td>
<td>8.01%</td>
</tr>
<tr>
<td>Caffeinemark</td>
<td>32 bit</td>
<td>18.32%</td>
</tr>
<tr>
<td></td>
<td>64 bit</td>
<td>31.13%</td>
</tr>
<tr>
<td>Stanford</td>
<td>32 bit</td>
<td>14.91%</td>
</tr>
<tr>
<td></td>
<td>64 bit</td>
<td>29.43%</td>
</tr>
</tbody>
</table>

### Android N vs. Android M

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Mode</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmarksgame</td>
<td>32 bit</td>
<td>18.90%</td>
</tr>
<tr>
<td></td>
<td>64 bit</td>
<td>21.32%</td>
</tr>
<tr>
<td>Caffeinemark</td>
<td>32 bit</td>
<td>17.13%</td>
</tr>
<tr>
<td></td>
<td>64 bit</td>
<td>25.36%</td>
</tr>
<tr>
<td>Stanford</td>
<td>32 bit</td>
<td>14.91%</td>
</tr>
<tr>
<td></td>
<td>64 bit</td>
<td>29.43%</td>
</tr>
</tbody>
</table>

---

1 - Benchmark numbers have been collected on A53 Nexus 5X.
ART Talks @ LCU16

- **ART Talks:**
  - ART JIT in Android N
  - VIXL: A Programmatic Assembler and Disassembler for AArch32
  - Android Runtime Performance Analysis
Thank You

#LAS16
For further information: www.linaro.org
LAS16 keynotes and videos on: connect.linaro.org
Coming up next

- Building AOSP with clang master in CI (and stabilize it, including kernel)
- Open Source benchmarks
- Continue increasing upstream participation
- Continue LCR improvements
- Continue to reduce AOSP kernel patch backlog
- Continue HAL Consolidation efforts
- Next LTS kernel support for Android, and into AOSP
- …
- More projects being decided on during Connect
Thank You

#LAS16
For further information: www.linaro.org
LAS16 keynotes and videos on: connect.linaro.org