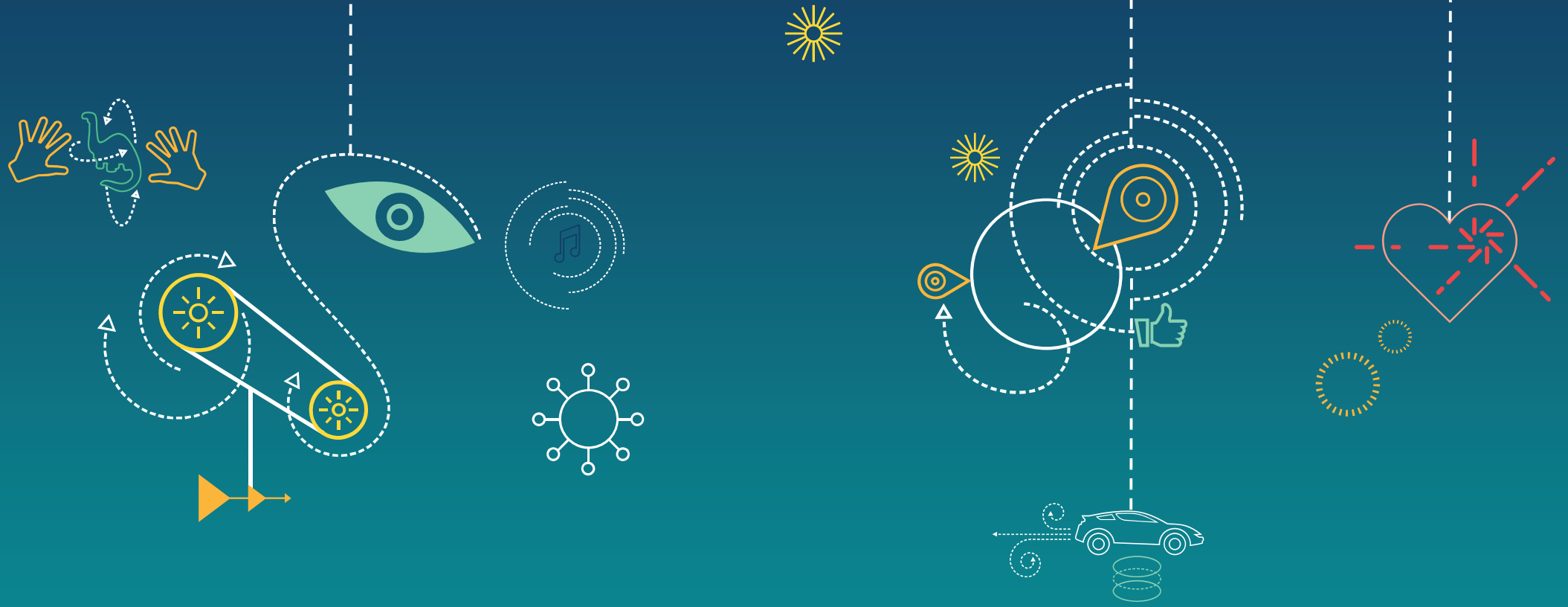
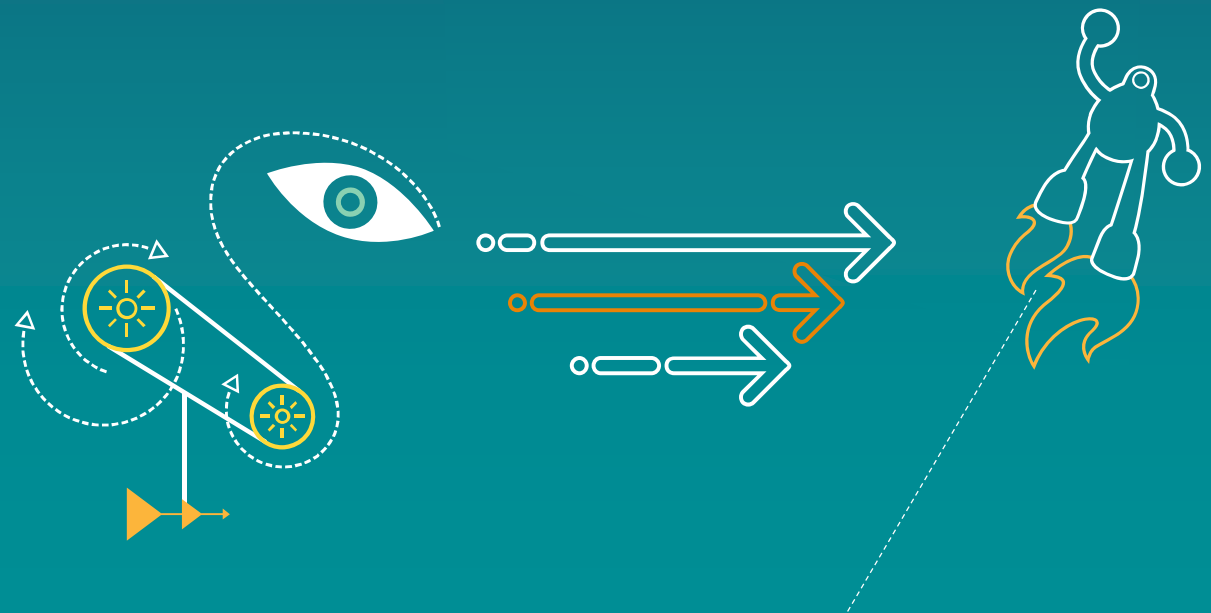


QUALCOMM®

Laura Abbott, Senior Engineer Qualcomm Innovation Center, Inc
October 16th, 2014

Ion and the DMA coherency model

QUALCOMM®



Purpose of this talk

Ion is doing many things wrong with coherency.

How can we fix them?

Not the primary purpose of this talk

- Constraint solving for devices
- Defining a stable ABI for userspace
- Reworking the Android graphics framework

What exactly is meant by coherency?

“Consistent memory is memory for which a write by either the device for the processor can immediately be read by the processor or device without having to worry about caching effects”

(Documentation/DMA-API.txt)

What exactly is meant by coherency?

Determination of what (if any) cache maintenance operations routines need to be performed on buffers to ensure correctness of data. These may be explicit where the goal is to only sync the cache or happen as part of another function.

What is Ion?

- Memory manager written for Android
- Primarily for graphics, other uses as well
- Generally allocator, dma_buf exporter

Ion terminology

- Ion cached buffers = non consistent memory
- Ion uncached buffers = consistent memory

Ion terminology

- Carveout heap – Region of memory permanently removed from the buddy allocator
  - Synced at creation time with dma_sync_sg_for_device (No DMA map first)
- System heap – Allocates memory via alloc pages
  - Contains a page pool of uncached, pre zeroed pages which don’t require syncing
- CMA heap – Allocates memory via dma_alloc_coherent
  - Method to associate device with a heap
  - Non-coherent allocation disallowed

Ion cached (non coherent) memory

What's it look like?

- Carveout heaps sync at free time
- CMA heaps disallowed
- System heap syncs right after allocation time

Ion cached (non coherent) memory

Mechanisms for keeping in sync

- Explicit sync ioctl (ION_IOC_SYNC)
- Faulting mechanism similar to mechanism described in dma_buf documentation

Ion cached (non coherent) memory

What's wrong?

- Using DMA sync APIs without map first
  - No guaranteed enforcement from explicit sync API
  - Faulting mechanism assumes map from somewhere
- Shouldn't need to sync at allocation time

Ion uncached (coherent) buffers

What's it look like?

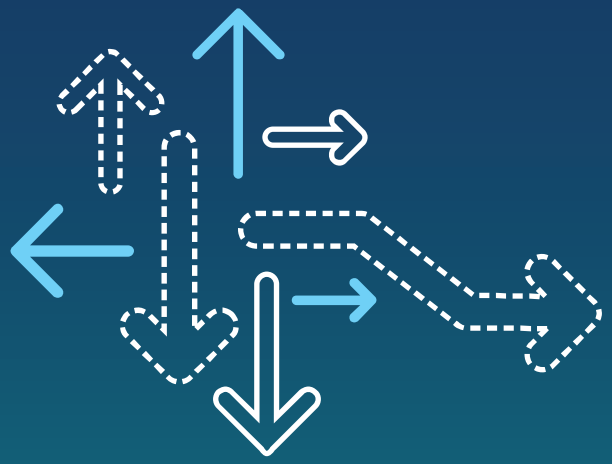
- Carveouts do nothing (synced at creation time)
- CMA does nothing
- System heap page pooling
  - If page in pool, take it. Otherwise, alloc from buddy allocator, zero, sync
  - On free, zero and place in pool
  - Shrinker drains pool as needed

Ion uncached (coherent) buffers

What's wrong

- Still using DMA sync APIs without a map for pooling
- CMA might actually be fully correct here!

Begin speculation



So what now?

Direction for the future of Ion. Unlikely candidates

- Ion as completely separate graphics framework
- Ion gets special blessing to use caching APIs as it is doing

Ion should stop trying to be the DMA layer

Just pull ion allocation methods into the DMA layer

- Ion heals already roughly correspond to different allocation methods
- Big issue: don’t know what device is being allocated for
  - Delayed freeing also throws this off

Ion should stop trying to be the DMA layer

Ion stops trying to do anything special with coherency

- Must call standard DMA APIs to get coherency
- Breaks the page pooling optimization for uncached pages
  - Page pooling is a must for performance reasons!

Other thoughts

- Patch from Hiroshi Doyu at nVidia for IOMMU mappings
  - http://lists.linaro.org/pipermail/linaro-mm-sig/2014-March/003768.html
- Sync ioctl to fences
- What to do about dma_buf ops?

Thank you

Follow us on:

For more information on Qualcomm, visit us at:
www.qualcomm.com & www.qualcomm.com/blog