World of Unified Memory Management

Tom Gall, John Stultz, Oct 28th 2013

Overview

Android Solutions	Upstream Solutions
Android Sync	Dmabuf-fences
Atomic Display Framework	KMS Atomic mode-setting & Nuclear page-flipping
ION	Dmabuf constraint solving & allocation

Android Sync vs dmabuf-fences

- Implicit vs explicit sync contracts
- Seemingly agreement on allowing for optional explicit sync arguments in the APIs
  - Likely trying to implement Android's Sync api on top of kernel-internal dmabuf-fences
- Next steps:
  - Maarten started looking at this, but may need assistance
  - Keep the discussion going
- Issues:
  - Lack of resources

ADF vs Atomic mode-setting/Nuclear pagefilp

- ADF: Simplified KMS-like framework.
  - Sort of an experiment to see what an ideal framework for Android would look like
  - Similar to upstream atomic mode-setting / nuclear pageflip work
  - Integrates Android sync points in the API
  - Presented as fodder for how KMS could be extended
- Next steps:
  - Continue pushing community and Android developers to collaborate on this.
  - Hopefully avoid forking display frameworks!
- Issues:
  - Lack of resources

ION vs Dmabuf constraint solving

- Discussed ION interface issues
- Upstream preferred solution is post-attach delayed allocation
  - Android developers have issues with this approach
- Likely to continue having separate interfaces, but hopefully will share underlying logic.
- Next steps:
  - Push ION into staging
  - Try to integrate ION's “heap” allocators in the dmabuf post-attach delayed allocation approach.
- Issues:
  - No open graphics drivers for proof of concept

Linaro

CONNECT

connect.linaro.org

More about Linaro: http://www.linaro.org/about/
More about Linaro engineering: http://www.linaro.org/engineering/
How to join: http://www.linaro.org/about/how-to-join
Linaro members: www.linaro.org/members