What is Android’s Hardware Composer?

- Determines the most efficient way to composite buffers with the available hardware
  - Overlay planes
  - GPU composition
  - Blit/2D engine
- Often device specific and written by the display hardware OEM
Hardware Composer 1.x

SurfaceFlinger

<table>
<thead>
<tr>
<th>prepare()</th>
</tr>
</thead>
<tbody>
<tr>
<td>set()</td>
</tr>
</tbody>
</table>

Hardware Composer

| vsync() |

Diagram: A sequence diagram showing the interactions between SurfaceFlinger and Hardware Composer, with functions `prepare()`, `set()`, and `vsync()`.
Key Differences between 1.x and 2.0

- Increases API functions from 12 to 43
- Adds support for HDR, color transform matrix, dataspaces, etc.
- Renames prepare() / set() to validate() / present()
- Replaces speculative fences with non-speculative fences
Hardware Composer 1.x Sync Fences

Renderer -> SurfaceFlinger
- send_buffers()
- previous_retire_fence && previous_release_fences []
- wait(previous_release_fences [])

SurfaceFlinger -> Hardware Composer
- prepare()
- set(acquire_fences [])
- retire_fence && release_fences []
- wait(acquire_fences [])

Hardware Composer -> Display Driver
- prepare()
- set(acquire_fences [])
- retire_fence && release_fences []

Display Driver -> Display
- vsync()
- write_buffers()
- wait(acquire_fences [])
- vsync()
- vsync()
Hardware Composer 2.0 Sync Fences

Renderer

SurfaceFlinger

Hardware Composer

Display Driver

Display

send_buffers() *

validate()

present_fence && release_fences []

wait(release_fences [])

wait(acquire_fences [])

display_driver

write_buffers()

displays

vsync()
Sources

- Hardware Composer 1.x and 2.0 Documentation
  - hardware/libhardware/include/hardware/hwcomposer.h
  - hardware/libhardware/include/hardware/hwcomposer2.h
  - hardware/libhardware/include/hardware/hwcomposer_def.h

- Coming soon: Nexus 9 Hardware Composer 2.0 Implementation
  - device/htc/flounder/hwc2/*
drm_hwcomposer

Sean Paul / Zach Reizner
Rectangle Separator

```c
struct DrmCompositionRegion {
    DrmHwcRect<int> frame;
    std::vector<size_t> source_layers;
};
```
GL Compositor

- uses separated regions directly
- generates a shader for each layer depth
- renders each rectangle region with one draw call
- no blending hardware used at all
- optimization: blending done within shader
- for layer import, uses NV hack: EGL_NATIVE_HANDLE_ANDROID_NVX
- for framebuffer import, uses standard EGL_ANDROID_image_native_buffer
- optimization: cache framebuffers using weakptr
Planner

- Introduced with Android N
- Planner runs every time the composition changes
- Platform register plan stages in priority order
- Plan stages map SurfaceFlinger layers to hardware planes
- After all stages finish, all layers should be mapped
Contributing to drm_hwcomposer

- Upstream source hosted on chromium.org gerrit
- External contributions welcome

https://www.chromium.org/android/contributing-to-drm_hwcomposer