Report: Unix Device Memory
Progress!

• Reached agreement on some key points:
  – Allocation requests consist of assertions/basic data (width, height, format, others via extension mechanism), and a list of usage descriptors
  – Allocation property arbitration is based on sets of supported capabilities (intersected) and sets of constraints (generally unioned, but exact merging logic baked into library).
  – Capabilities can be vendor-specific, constraint definitions are shared
• Reached agreement on some key points:
  – Capability sets are reported back to applications, can be serialized and shared across processes for incremental refinement.
  – Sorting of capability sets happens after filtering. Sorting is handled by drivers.
  – Allocation takes place after sorting and selecting a single capability set.
  – The new allocation API will be exposed via a centralized library that has userspace driver/vendor back-ends.
Unresolved

• Lot’s of stuff, but specifically
  – Not clear exactly how sorting happens yet. Various options identified.
  – How to tell app which state/layout transitions are needed, and how to perform them.
  – How formats are expressed
  – What type of surface handle is used
  – How (and if) devices are enumerated in new API
  – What kernel interfaces are used for allocation
Homework

• Reached a point where people need to start doing a bit of research then re-convene

• In the meantime
  – Make sure all allocation properties can be expressed as positive singular descriptors that can be properly filtered by intersection
  – Think of more interesting/corner use cases, especially those that might break current proposals.