Overview of TTM

Thomas Hellström
VMware Inc.

Outline

- What is TTM?
- Components
- Design philosophy
- Buffer objects
- Pros – Cons
- Future wishlist

What is TTM?

- TTM is short for “Translation Table Maps”, which refers to the initial functionality of transparently handling data that could be flipped in- and out of a Translation Table / GART
- Now a set of utilities that can be used to manage buffer objects in device / system memory as well as locking, mapping, accounting and execution management.
- Does not come with a user-space interface. Can , for example be interfaced with GEM.

Components

- Buffer objects, utilizing device-, GART- or system memory.
- Accounting and paging – buffer objects are paged out on a global basis.
- User-space mapping utilities.
- Execution utilities.
- User-space object utilities
- Locking utilities

Design philosophy

- Aggressively utilize available resources – free on demand.
- Fine grained locking.
- Simplify user-space sub-allocation

Buffer objects

- Each TTM device supports a driver-defined number of memory regions for data placement. (Device memory / GARTs).
- Each memory region has its own LRU list for buffer eviction.
- Swap LRU list is global.
- TTM currently supports only coherent memory. Achieves this by write-combined / uncached CPU mappings when required.

Buffer objects – ct'd

- Buffer objects can be pinned in current location (root only). Scanout- and capture buffers.
- Buffer objects can be “reserved” from kernel space. Acts like taking a mutex + removing from LRU lists. Need to be reserved when moved.
- User-space maps are persistent. Object may move while user-space writes to it. Page tables rewritten.
- In - kernel maps are temporary (reserved objects) or persistent (pinned objects).

Pros - Cons

- Well established / tested: Nouveau – Radeon – Vmwgfx – (VIA)
- Fine-grained locking
- Set of utilities
- Supports user-space sub-allocation

- Fine-grained locking
- Complex API / Code
- Documentation

Future wishlist

- Asynchronous memory management for devices with long command queues.
- Read / Write dirty – tracking.

Supporting shared buffer objects

- Memory regions shareable between TTM devices.
- User-space mapping.
- Non-coherent memory – Needs API to flush CPU caches.