glTF Texture Transmission Extension

David Wilkinson

Game Engineering Group, AMD
Chair, 3D Formats Texture Transmission TSG

Whirlwind Agenda

- Introduction to the 3D Formats Texture Transmission TSG
- Current methods for GPU texture compression and distribution
- Introducing the ‘Universal Format’
- Proposed KHR_texture_transmission extension
- Call To Action
- Snacks and Nap-time

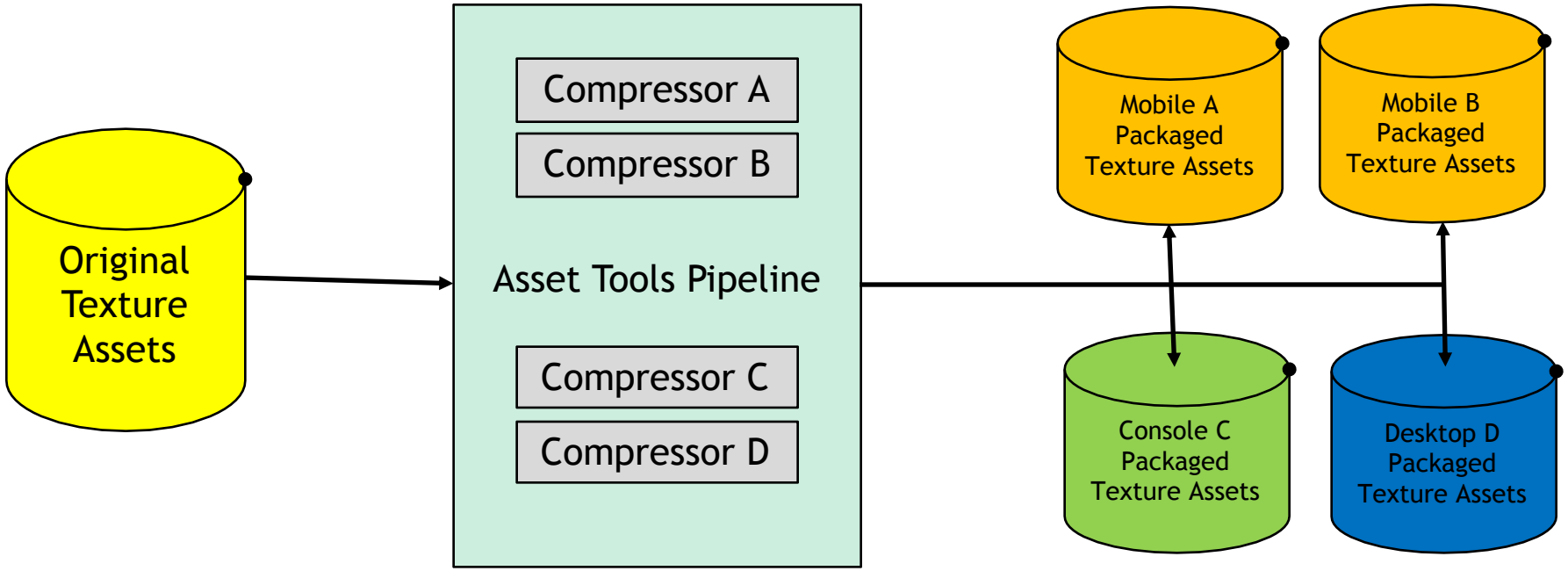
Introduction - Texture Transmission TSG

- glTF provides means to efficiently transmit 3D scene data, but large texture assets are challenging. No jpg or png for scene textures!
- Most GPUs (mobile and discrete) support hw decode of block-compressed textures
- But, these formats are not always compatible across GPU device classes and platforms
- How to transmit texture data via glTF in a way that all target hw can use?
- Not a trivial problem to solve
- Current design focuses largely around Binomial's Basis solution, which they have kindly donated to Khronos - http://www.binomial.info
- Top tier ISVs, IHVs, and Independent Contributors all working together on a solution

GPU Texture Compression Formats

GPU format	Source data	Pixel Encoding	Typical Device Support
PVRTC	Three-channel color with optional alpha	Three or four color channels (2bpp, 4bpp), with 0 - 3 bit alpha	Mobile
ETC1	One to four channel multipurpose	Three channel color (4bpp)	Mobile
BC1	Three-channel color with alpha channel	Three color channels (5 bits:6 bits:5 bits), with 0 or 1 bit(s) of alpha	Mobile Desktop GPU
BC2	Three-channel color with alpha channel	Three color channels (5 bits:6 bits:5 bits), with 4 bits of alpha	Some Mobile Desktop GPU
BC3	Three-channel color with alpha channel	Three color channels (5 bits:6 bits:5 bits) with 8 bits of alpha	Mobile Desktop GPU
BC4	One-channel color	One color channel (8 bits)	Mobile Desktop GPU
BC5	Two-channel color	Two color channels (8 bits:8 bits)	Mobile Desktop GPU
ASTC	Three channel color with optional alpha	2bpp, 4bpp, 8bpp	Mobile Desktop GPU
BC6H	Three-channel high dynamic range (HDR) color	Three color channels (16 bits:16 bits:16 bits) in "half" floating point*	Desktop GPU
BC7	Three-channel color, alpha channel optional	Three color channels (4 to 7 bits per channel) with 0 to 8 bits of alpha	Desktop GPU

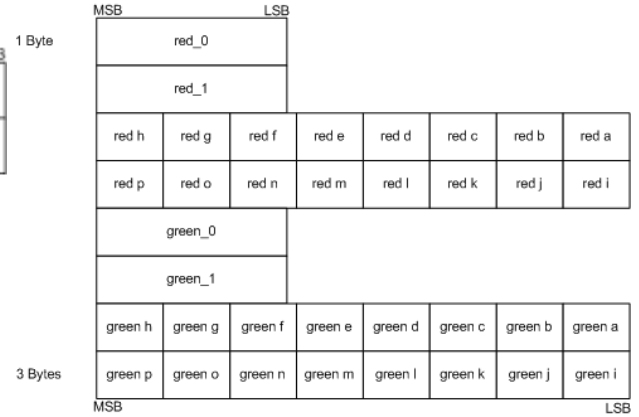
Current Texture Tool Workflows



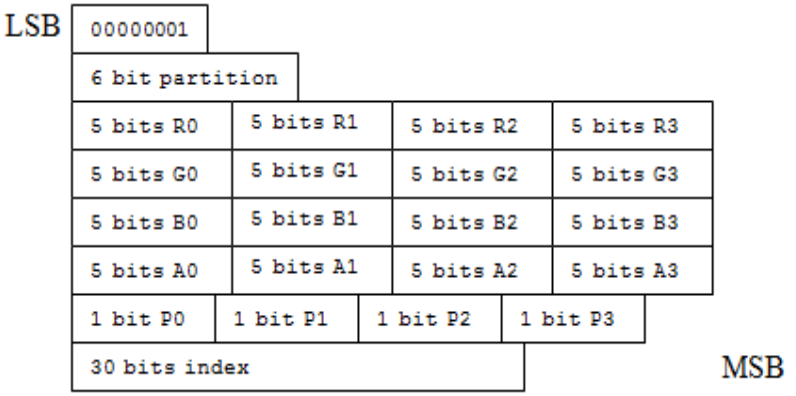
Introducing the Universal Format

- Each GPU format is different…
- But , each format's structure share common elements

BC1 Block Format



BC5 Block Format



BC5 Mode 7 Block Format

Introducing the Universal Format

- **Common Features**
  - All formats have at least two color endpoints
  - All formats have a set of selectors (color interpolants) for each pixel in the block
  - Some have optional alpha flags and bit values
  - Advanced formats have extra endpoints, flags, multiple precision and multiple partitions

- **What if we reverse engineer all formats and find all common traits?**
  - Then combine them into one format that can represent them all
  - Such a format would then be considered a ‘universal’ format
  - If a device requires, say, BC5 from a universal format – then just ‘transcode’ the compressed source to a BC5 representation

- **The Texture Transmission extension will support such a Universal Format**
  - Based on tried and tested shipping version of Binomial Basis Universal Format
  - Basis format and sample transcoders will be donated by Binomial to Khronos

Universal Format – Encode and Transcode

- **Universal Format representation can implicitly store any combination of GPU formats.**
  - Special encoders are required to create the internal representation and dictionaries, mapping tables etc.
  - The texture transmission extension would provide sample implementations
  - All open-source, developers can make their own
  - There are also commercial encoders available providing higher quality, tunable parameters, GUI interface etc.

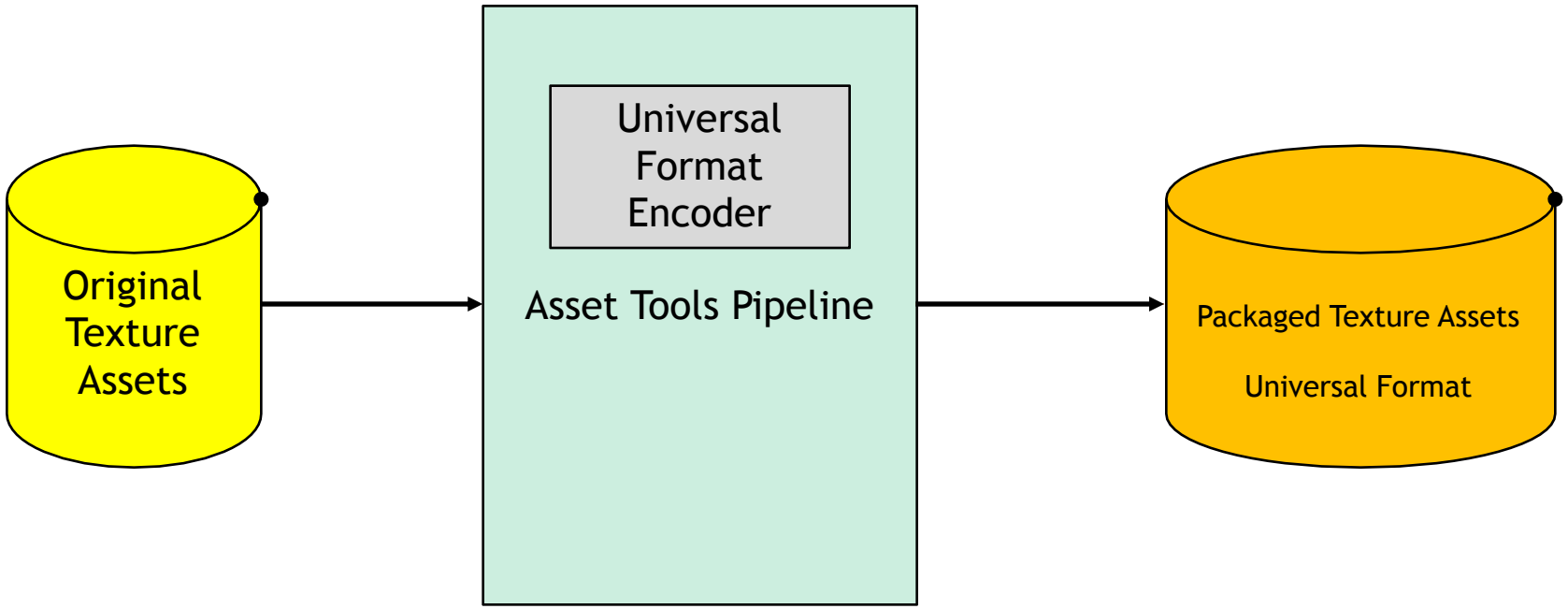
- **On the target, the universal format can be transcoded into any desired format**
  - This is done via transcoding
  - Transcoding is different from ‘decoding’ in the compression sense
  - Transcoding is real time, and converts *directly* to the desired format very fast
  - So fast, that most transcoders can be written in Javascript for web use
  - Transcoding can be done on the target CPU *or* GPU
  - Binomial Basis format implements all this very efficiently

Universal Format and Compression

- **Due to the way the source texture data is encoded into a Universal Format representation, there are opportunities to apply compression**
  - Creating the the universal representation uses VQ and Clusterization to optimize endpoints and selectors
  - This, in addition to other entropy reduction methods can reduce the format to very high compression rations
  - Potentially matching JPG in some cases

- **An optional lossless encoder will be supplied that can further compress texture data**
  - Options will include a standard LZ and rANS decoder
  - Both GPU and CPU versions with be provided
  - Proprietory codecs can also be used
  - Tools will be provided with the extension to optimize source texture data prior to lossless compression
  - Allows for JPG-level textures to be transmitted and decoded directly on GPU for desktop platforms

Texture Workflow Using Universal Format



Supported Texture Types

- **The Universal Format encoders and transcoders work well with many types of texture data**

- **The following have all been tested to work with Binomial Basis**
  - Photographs
  - UI elements, Texture arrays, slices, mipmaps, cubemaps
  - Video textures
  - Animated textures
  - Lightfields, Normal maps, materials etc.

- **And other types are in the pipeline**
  - HLODS
  - 3D Volumes

Supported Features

- **As of now, the following features are in shipping or development stages:**
  - Basis Universal Texture Codec for Windows/OS/Linux is shipping from Binomial
  - CPU transcoders available
  - GPU transcoders currently in development
  - Supports BC1-BC5 and ETC transcoding from the same compressed source
  - Supports RGB, RGBA and Alpha-only textures
  - Support for arbitrary 2D arrays
  - BC6H and BC7 transcoders currently in development
  - Video Texture formats currently in design phase
  - Progressive reconstruction on decode currently in design phase

KHR_texture_transmission Extension

- **This extension will allow for glTF authoring tools to import and export scenes with compressed universal format textures**
  - The format will be built upon the Basis file format and transcoders that Binomial are donating to Khronos
  - Other members of the TSG are welcome to submit design proposals for new, or modified features, supported formats and codecs

- **The extension will provide the following functionality**
  - Universal Formats (low precision and high precision versions)
  - Ability to choose which formats are encoded into the UF texture (smaller footprint)
  - Optional LZ and rANS lossless encoders will be provided to compress texture data down to JPG-level sizes
  - Open-source transcoders and lossless decoder implementations will be provided on the extension github repo
  - All transcode/decompression operations will have CPU and GPU based sample implementations
  - Support for transmission of proprietary compression schemes

Call To Action

- **We Need You!**

  - The extension is still in the design phase, and we would be very keen to have more members join us in developing the extension

  - Sign up for Khronos 3DFormats group , go to www.khronos.org for details on membership and how you or your company can become involved in the glTF design process and the Texture Transmission Task Group

Thank You!