Unreal Engine 4: Mobile Graphics on ARM
CPU and GPU Architecture

Ray Hwang, Segment Marketing Manager, ARM
Jack Porter, Engine Development Lead, Epic Games Korea
Hessed Choi, Senior Field Applications Engineer, ARM
Agenda

- Programming for ARM®v8-A Technology
- ARM Mali™ GPU Architecture
- Unreal Engine 4 Case Study: Moon Temple
- Enlighten in Unreal Engine 4
Programming for ARMv8-A Technology

Ray Hwang
Segment Marketing Manager, ARM
ARM Architecture Evolution

1995

ARMv4
ARM7TDMI

ARMv5
ARM926EJ

ARMv6
ARM1176

ARMv7-A/R
Cortex®-A9

ARMv8-A
Cortex-A57

2005

NEON™ Adv SIMD

2015

Virtualization

A32+T32 ISAs
Including:
- Scalar FP (SP and DP)
- Adv SIMD (SP Float)

A64 ISA
Including:
- Scalar FP (SP and DP)
- Adv SIMD (SP+DP Float)

CRYPTO

Cryptographic

CRYPTO
ARMv8-A AArch32
Maintaining compatibility

- AArch32 maintains full-compatibility with ARMv7 while addressing emerging software trends

- AArch32: evolution of 32-bit
  - Enhanced floating point support (IEE754-2008)
  - Ideal for concurrent programming
    - C11, C++ 11, Java5
  - More efficient, high-performance thread-safe software
  - Cryptography support (AES, Sha-1, Sha-256)
# ARMv8-A Architecture

Design: Designed for efficiency

<table>
<thead>
<tr>
<th>Design</th>
<th>Why it Matters</th>
</tr>
</thead>
<tbody>
<tr>
<td>64-bit architecture</td>
<td>Efficient access to large datasets</td>
</tr>
<tr>
<td>Increased number and size of general purpose registers</td>
<td>Gains in performance and code efficiency</td>
</tr>
</tbody>
</table>
| Large Virtual Address Space                 | 1. Applications not limited to 4GB memory  
2. Large memory mapped files handled efficiently         |
| Efficient 32-bit/64-bit architecture        | 1. Common software architecture (phone, tablet, clamshell)  
2. A single software model across the entire portfolio      |
| Double the number and size of NEON™ registers | Enhanced capacity of SIMD multimedia engine                                                             |
| Cryptography support                        | 1. Over10x software encryption performance  
2. New security models for consumer and enterprise       |
Multi-core ARM big.LITTLE™ Technology

Taking advantage of parallelism

- Platform trending toward multi-cores
  - Single thread performance improvements diminishing
  - Thermally constrained use cases are now commonplace
  - Production differentiation via different CPU combinations

- Modern OSs are supporting multi-core

How to exploit parallelism….

In the core
- ARM NEON tech/SIMD

API-level parallelization
- OpenMP®, Renderscript, OpenCL™, etc.

Multi-thread programming
- Never easy, but increasingly necessary
Mali™ GPU Architecture

Hessed Choi
Senior FAE, ARM
Mali GPU High-Level Architecture

A breakdown of the Mali-T880

- Distributes tasks to shader cores
- Efficient mapping of geometry to tiles
- Addresses translation and protection
- Maintains cache coherency between different processors
- Up to sixteen shader cores
- Configurable cache shared among all shader cores
- Efficient mapping of geometry to tiles
- Advanced Tiling Unit
- Memory Management Unit
- Efficient mapping of geometry to tiles
- Maintains cache coherency between different processors
- Configurable cache shared among all shader cores
- Efficient mapping of geometry to tiles
- Advanced Tiling Unit
- Memory Management Unit
Shader Core Architecture

- Compute Thread Creator
- Rasterizer
- Triangle Setup Unit
- Tiler Data Structures
- Early Z
- Thread Execution – “Tri Pipe”
- Thread Issue
- Thread Completion
- Compute Data and Results
- Textures
- Z/Stencil Buffer
- Late Z
- Blender
- Tile Buffers
- Frame Buffer
- Texturing
- Arith / LUT / Branch
- Load / Store / Varying
- Reg file
- Reg file
Tri-pipe Architecture

- Unified shader architecture
  - Fragment and vertex shaders
  - Compute shaders
  - Very high throughput graphics

- Multiple parallel pipelines
  - Three low-latency arithmetic pipes (in case of Mali-T880)
  - 256 simultaneous threads
  - Low-latency for computation
Deferred Shading

- Popular technique in PC and console games
  - Very memory bandwidth intensive
  - Traditionally not a good fit for mobile
The Tilebuffer

- Mali-T600 Series GPU
  - Tile-based rendering

- 16x16 tile size
  - Fast on-chip memory
  - 16 bytes of per-pixel color data
  - Raw bit access

- More recent GPU architectures allow more flexible tile sizes and open up more per-pixel color data
Exposing the Tilebuffer

- Shader Framebuffer Fetch
  - Access previous fragment color, depth and stencil
  - Programmable blending, soft particles, etc.

- Shader Pixel Local Storage (PLS)
Pixel Local Storage (PLS)

- Exposed as EXT_shader_pixel_local_storage

- Per-pixel scratch memory available to fragment shaders
  - Automatically discarded once a tile is fully processed
  - No impact on external memory bandwidth

- Shader declares an interface block of PLS memory
  - Re-interpret PLS between different passes
  - Can have separate input and output views
  - Independent of framebuffer format
Pixel Local Storage – Shader example

```glsl
__pixel_localEXT FragDataLocal
{
    layout(r32f) highp float_value;
    layout(r11f_g11f_b10f) mediump vec3 normal;
    layout(rgb10_a2) highp vec4 color;
    layout(rgba8ui) mediump uvec4 flags;
} pls;
```

- See the extension spec for more information!
  - [https://www.khronos.org/registry/gles/extensions/EXT/EXT_shader_pixel_local_storage.txt](https://www.khronos.org/registry/gles/extensions/EXT/EXT_shader_pixel_local_storage.txt)
  - [http://malideveloper.arm.com](http://malideveloper.arm.com)
Pixel Local Storage - Pipelines

- Rendering pipeline changes slightly when PLS is enabled
  - Writing to PLS bypasses blending

- Note
  - Fragment order
  - Fragment tests still apply
  - PLS and color share the same memory location
Why Pixel Local Storage?

- An alternative approach is to use multiple render targets (MRT) with framebuffer fetch
  - …if the driver can prove that render targets are not used later, it can avoid the write-back

- PLS is more explicit than MRT
  - Harder for the application to get it wrong
  - Driver doesn’t have to make guesses

- PLS is more flexible
  - Re-interpret PLS data between fragment shader invocations
  - Not limited to OpenGL® ES 3.x framebuffer formats
Unreal Engine 4

Jack Porter
Engine Development Lead, Epic Games Korea
Compress, Compress, Compress!

- **ASTC = Adaptive Scalable Texture Compression**
  - Texture compression standard developed by ARM, adopted by Khronos
    - KHR_texture_compression_astc_ldr for OpenGL ES and OpenGL®
  - Increased quality and fidelity at low bit-rates
  - Expansive range of input formats offers complete flexibility
    - Choice of base format, 2D and 3D plus addition of HDR formats
Compression in the Pre-ASTC World

Input Color Formats

HDR RGB+A
HDR RGBA
HDR XY+Z
HDR RGB
HDR X+Y
RGB+A
RBGA
XY+Z
RGB
HDR L
X+Y
LA
L

All Major Players

ETC, BC1
ETC, BC4
ETC, BC2
ETC, BC3, BC7
ETC, BC5
BC6
BC7

Compressed bits/pixel

Input bits/pixel

1 2 3 4 5 6 7 8

ETC, BC5
BC7
ETC, BC2
ETC, BC3, BC7

ASTC Choices

Input Color Formats

- HDR RGB+A
- HDR RGBA
- HDR XY+Z
- HDR RGB
- HDR X+Y
- RGB+A
- RBGA
- XY+Z
- RGB
- HDR L
- X+Y
- LA
- L

Compressed bits/pixel

- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8

Input bits/pixel

- 8
- 16
- 24
- 32
- 48
- 64

All ASTC
ASTC for Mobile Games

- ASTC is widely supported by all major hardware vendors
  - It’s free to use

- Finally a good texture format that can work everywhere!
  - Avoids separate SKUs per hardware manufacturer: PVRTC, ATC, DXT, …
  - `<supports-gl-texture android:name="GL_AMD_compressed_ATC_texture" />`

- Support for ASTC is also required by Google’s Android Extension Pack
  - GL_ANDROID_extension_pack_es31a
Build, pack, and package your game for the Android (ASTC) platform.
Game Texture Comparison

- 2048x2048 RGB Normal Map, with mips – 17 MB uncompressed
Game Texture Comparison

- Same texture – zoomed in for Truth
Unreal Engine 4 – Framebuffer Fetch

- Unreal Engine 4.9 makes use of Mali’s support for efficiently reading the existing color and depth values while rendering for a number of rendering features
  - GL_ARM_shader_framebuffer_fetch
  - GL_ARM_shader_framebuffer_fetch_depthStencil
Unreal Engine 4 Demo: Moon Temple

- Made specifically for ARM
- Unreal Engine 4

Goals:
- 64-bit Android
- ASTC
- PLS
Unreal Engine 4 – Pixel Local Storage

- Read & write custom data for each pixel
- E.g. depth and HDR color
- Blend particles softly against the background
Unreal Engine 4 – Pixel Local Storage

PLS Disabled

PLS Enabled
Moon Temple Demo
Enlighten in Unreal Engine 4

Ray Hwang
Segment Marketing Manager

Geomeric
An ARM company

The Architecture for the Digital World®
The Global Illumination Challenge

- Direct illumination only takes into account light coming directly from the source
- Global illumination computes the way this light is absorbed, reflected, scattered and refracted across various surfaces in a scene

- Traditional methods to achieve this effect:
  - Addition of light sources
  - Offline lightmap bake
The Enlighten Solution
The industry’s most advanced global illumination technology

- Enlighten pre-bakes the location of static geometry in scene

- Everything else is dynamic at runtime
  - Move lights
  - Change materials
  - Move dynamic objects

- Enlighten computes indirect lighting in real time. Benefits include:
  - Higher quality
  - Faster workflow
  - Differentiated gameplay
  - High scalability
Higher Quality in Unreal Engine 4
Leading Titles use Enlighten

- Battlefield Hardline, EA
- Battlefield 4, EA
- Plants vs Zombies: Garden Warfare, EA
- Need for Speed: The Run, EA
- Dragon Age: Inquisition, EA
- Quantum Conundrum, Square Enix
- Lords of the Fallen, CI Games
- Republique, Camouflaj
- Secret Ponchos, Switchblade Monkeys
- PollenVR, Mindfield Games
Available for Any Engine or Gaming Platform

- Pre-integrated into Unreal Engine
- Licensable as a standalone SDK for proprietary engines
- The real-time GI solution for Unity 5
- The runtime is highly optimised and runs on many platforms
  - **PC**
    - Windows, Mac OS X, Linux
  - **Console**
    - Xbox 360, Xbox One, PlayStation 3, PlayStation 4, PlayStation Vita, Wii U
  - **Mobile**
    - Android, iOS, Windows Phone
To Find Out More…

- Did you check with ARM booth, to find out more about ARM Mali, ASTC, Enlighten, etc.?

- If not, you still have chances
  - Mali Developer Center ([http://malideveloper.arm.com/](http://malideveloper.arm.com/)) : demos, tools, SDKs, tutorial & guides, sample codes, etc.
  - Contact our experts at malidevelopers@arm.com : you can get help when you encounter any problems
  - Forum ([http://community.arm.com/groups/arm-mali-graphics](http://community.arm.com/groups/arm-mali-graphics)) : ask questions, discuss, debate, search for Q&As, read blogs, and do many other things
Thank You

The trademarks featured in this presentation are registered and/or unregistered trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved. Any other marks featured may be trademarks of their respective owners.