Low power GPUs
a view from the industry

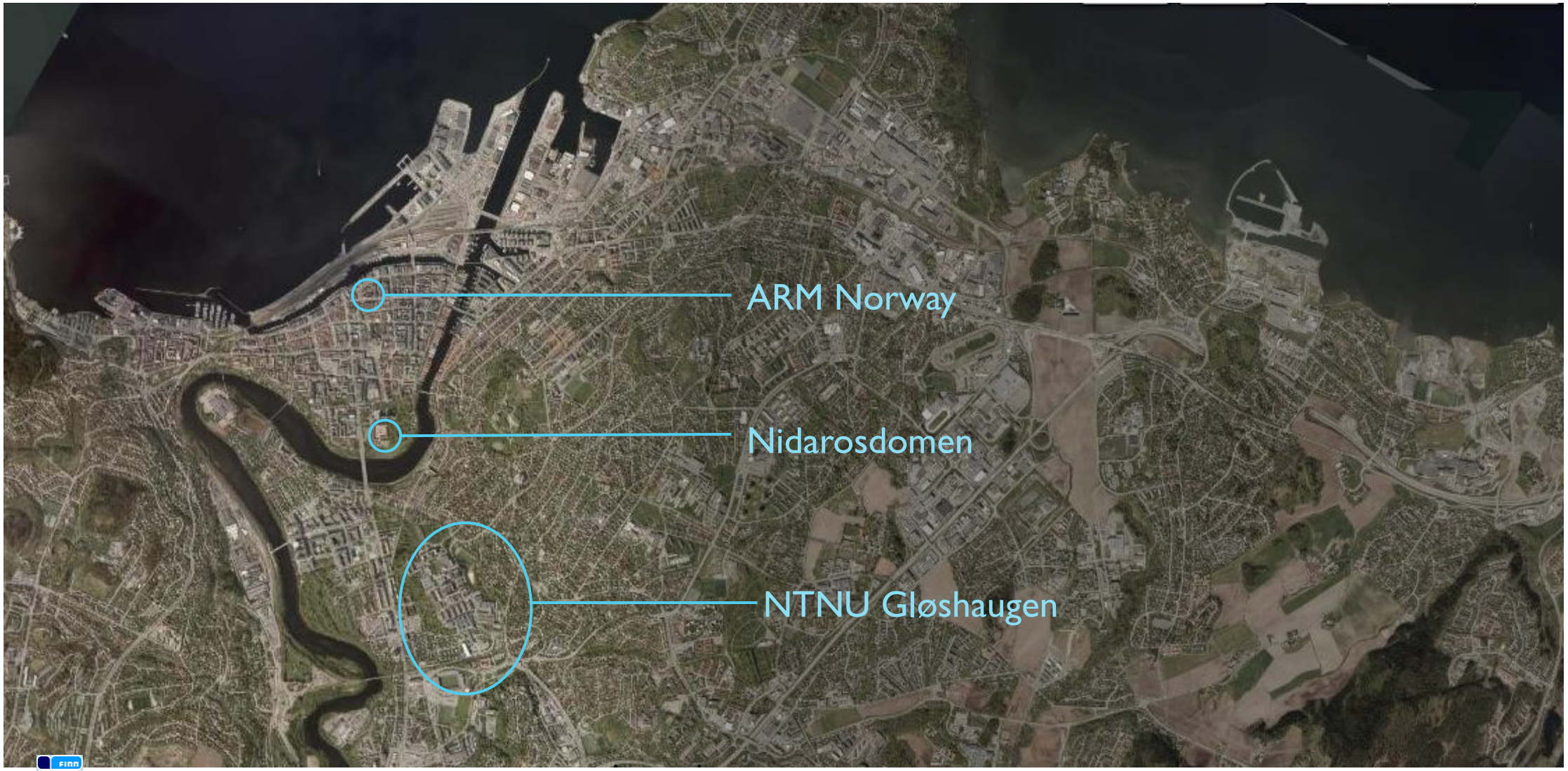
Edvard Sørgård

ARM in Trondheim

- Graphics technology design centre
- From 2006 acquisition of Falanx Microsystems AS
- Origin of the ARM® Mali™ GPUs

- Main activities today
  - GPU HW development
  - Driver SW development
  - Systems (ASIC/FPGA)
  - Performance analysis

ARM in Trondheim



The Need for Power Efficiency

- Performance requirements are endless...
- Thermal is the #1 performance barrier
  - Easy to make high performance GPUs
  - Hard to reach that performance on mobile power
- Energy efficiency → more performance
- The best energy efficient performance is achieved by addressing the entire system design

ARM

The Power Challenge is Universal

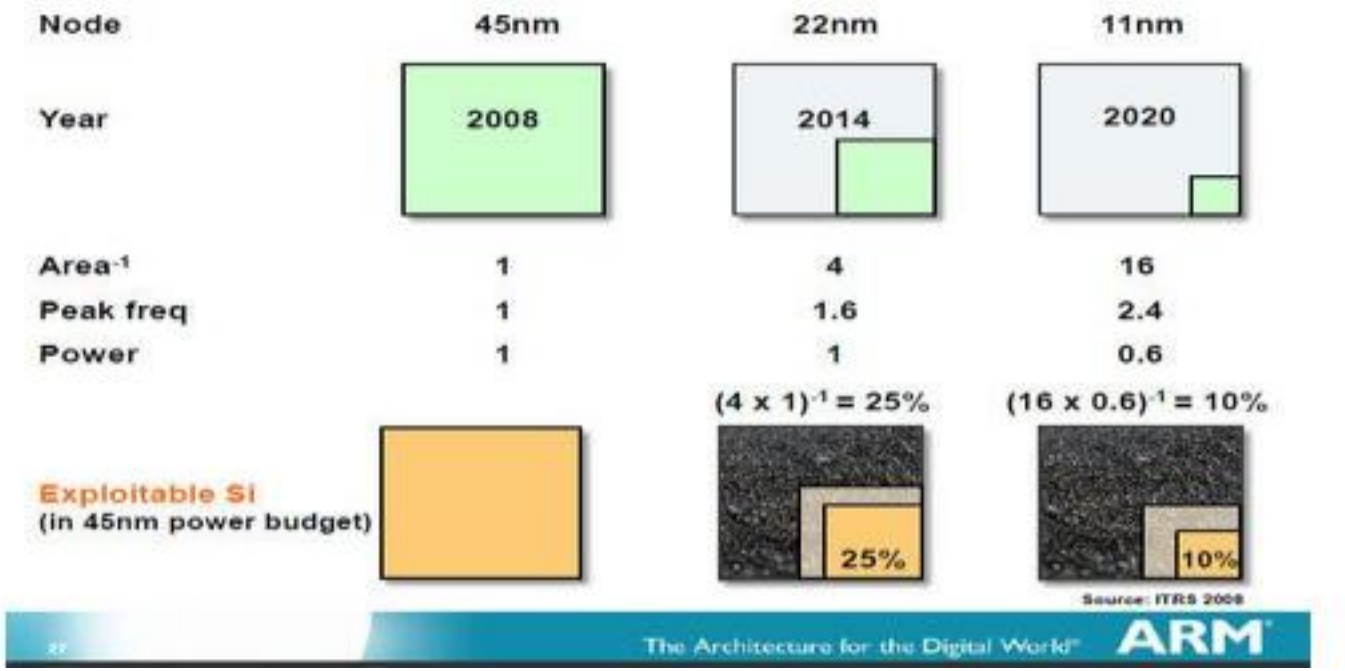
- Every performance device category is now limited on Power
- Notice
  - For human touched devices thermal power is the limit
  - For others supply constraints are likely the limit

- Example power limits/budgets
  - Mobile: max ~1W chip dissipation
  - Tablet: max ~3W chip dissipation
  - Ultraportable: max ~7W chip dissipation
  - Laptop: max ~35W chip dissipation
  - Desktop: max ~150W chip dissipation
  - Server rack: max ~10kW dissipation
  - Data centre: max ~100MW supply

ARM

Dark Silicon – The Problem

- Silicon process nodes continue to shrink area
- But the new area can't be powered inside previous budgets!
- E.g. an 11nm chip with a 1W power budget can only run 10% of the Si area compared to a 45nm chip with the same budget
- 90% of a same sized chip will have to be off i.e. be *dark*



Dark Silicon – The Challenge

- Increasing performance turns into a pure quest to

Improve Energy Efficiency

- A specialized version of this quest is optimizing for a power budget that is low

ARM

Quest Toolbox

- What are our main tools to improve energy efficiency?

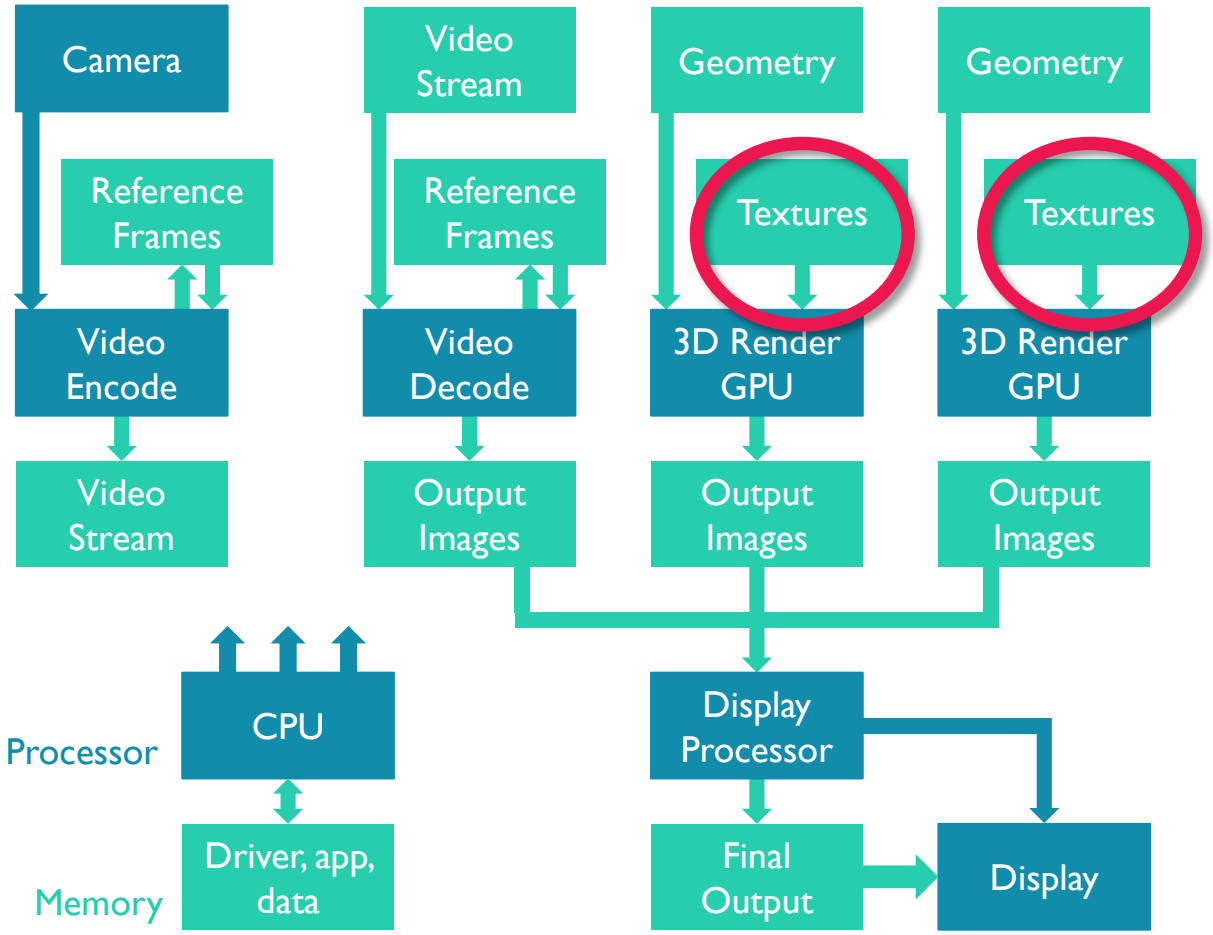
1. Architecture and micro-architecture
   - Smarter algorithms and machines structures
2. Implementation
   - More efficient physical process nodes
   - Better tools and implementation know-how
3. Use model / Software
   - Better management software
   - New APIs, improved application interaction
   - Compiler technology

ARM

Texture Compression

- Texture access is a big user of bandwidth
  - Up to 90% of content memory reads in 3D content are texture reads
- Texture compression is the solution
- ARM texture compression is best in class

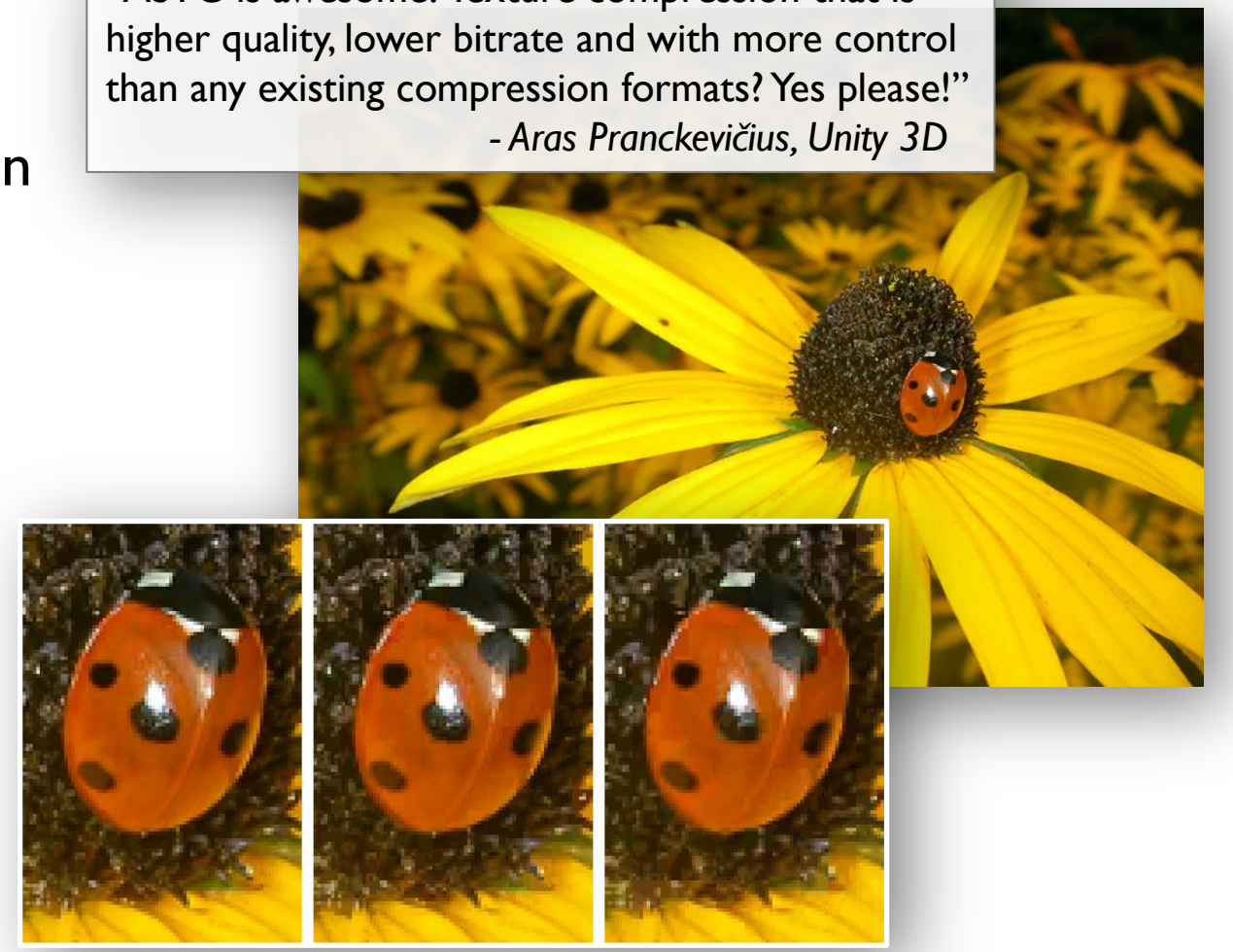
See the ASTC demo video from CES 2014 at
http://community.arm.com/groups/arm-mali-graphics/blog/2014/01/21/arm-mali-gpu-technology-at-ces--whats-new-in-2014



Texture Compression - ASTC

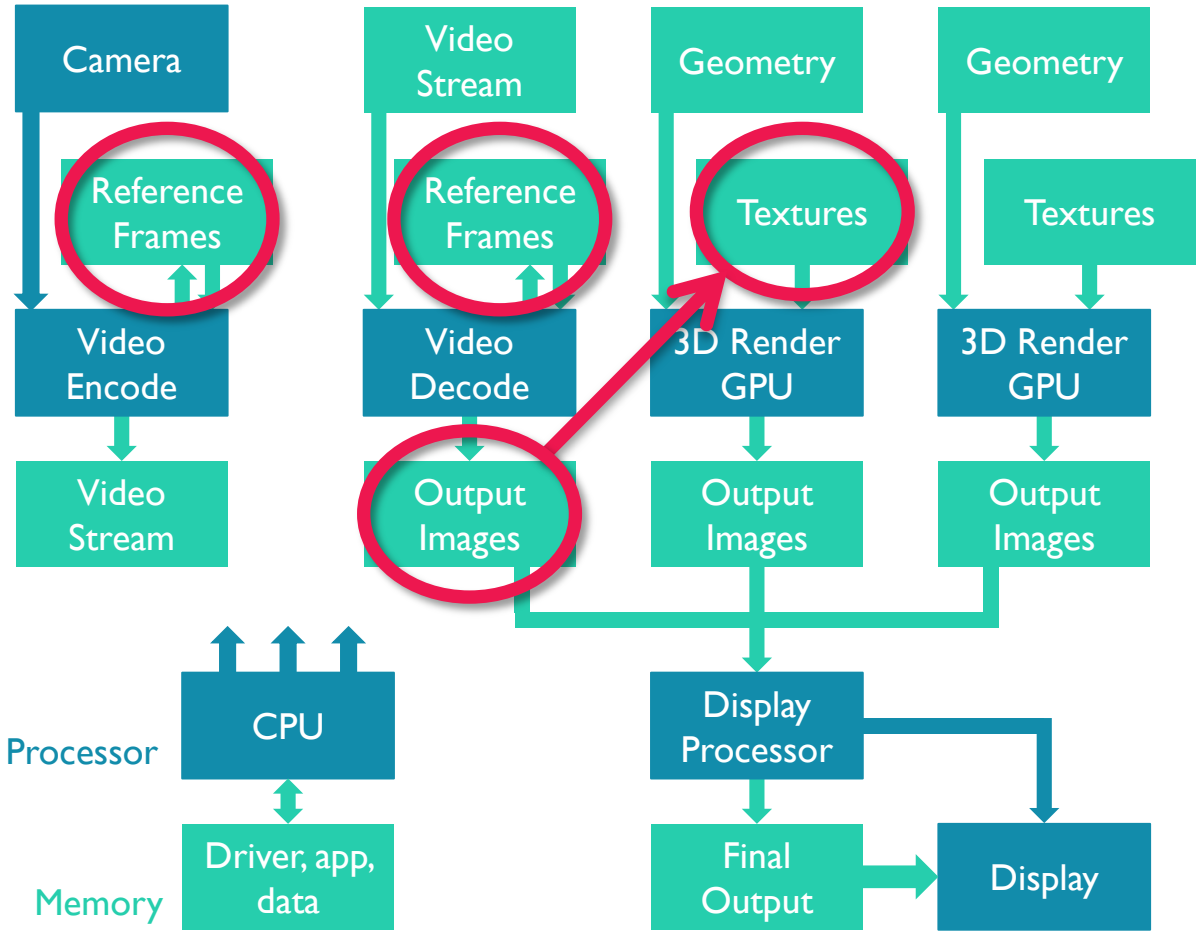
- ARM® Mali™ GPUs support ARM's Adaptive Scalable Texture Compression (ASTC)
  - Better reduction in texture size
  - Better quality at the same time
- Better flexibility
  - More choice of pixel formats
  - More choice of bit rates
  - Allows content developers to choose best tradeoff of size v quality
- Now an industry standard

"ASTC is awesome! Texture compression that is higher quality, lower bitrate and with more control than any existing compression formats? Yes please!"
- Aras Pranckevičius, Unity 3D



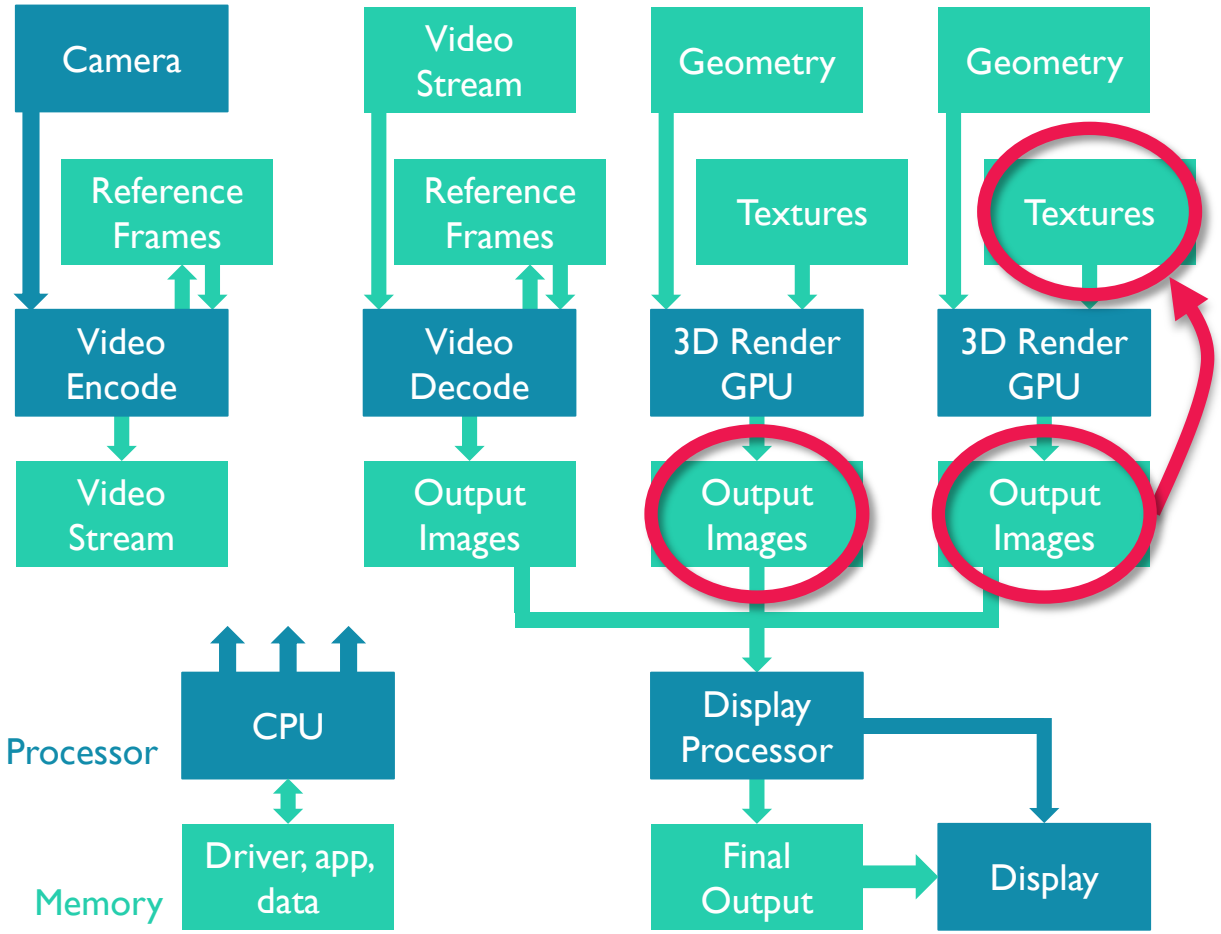
Frame Compression - AFBC

- ARM® Mali™ Video processors support ARM Frame Buffer Compression (AFBC)
- Lossless compression for frame buffers
  - Invisible to user
  - 40% bandwidth reduction in typical content
- Internal and external use cases
  - Reference frame bandwidth reduction
  - Compressed transfer from video processor (VPU) to GPU



Frame Compression - AFBC

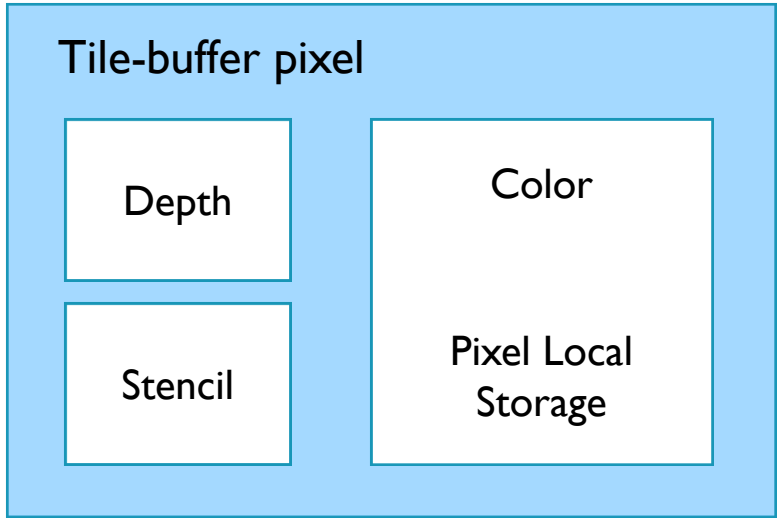
- ARM® Mali™ GPUs also support AFBC
  - Useful in reducing output bandwidth
  - Also available for GPU input as a texture
- Useful for various applications
  - VPU-GPU transfer
  - Compression of sensitive textures
- Particularly useful in G-buffer rendering
  - Compressed render-to-texture
  - Texture is re-read by the GPU later



ARM® Mali™ Extensions

Efficient deferred shading

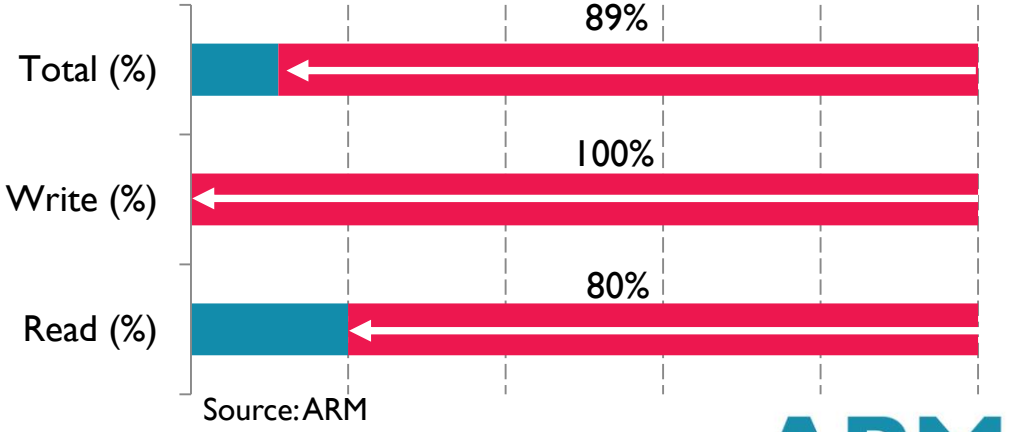
- Shader Frame buffer Fetch (OpenGL® ES 2.0)
  - Reading of on-chip frame buffer color, depth and stencil values
- Shader Pixel Local Storage (OpenGL® ES 3.0)
  - Reading and writing the current pixel's data locally
  - Persistent throughout the lifetime of the frame buffer
- Deferred shading with Mali extensions
  - Avoid multiple render targets and rendering passes
  - Data remains on-chip saving power and bandwidth



Shader Frame Buffer Fetch

Shader Pixel Local Storage

Bandwidth savings with Mali deferred shading



Final Key Take-Aways

- The power limit is the #1 barrier to continued improvement in the industry
- Luckily there is a broad set of solution layers
  - This will help keeping the momentum high
  - GPUs will continue to deliver significantly improved efficiency

- But the forces of economics are strong
  - The best solutions will have to optimize within both *power, cost and time* envelopes
  - If you want to benefit make sure you are aligned or heard

ARM

Thank You!
Questions?

ARM