HOT CHIPS

KHRONOS GROUP

SOC Programming Tutorial

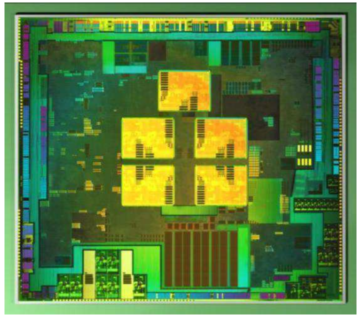
Hot Chips 2012

Neil Trevett
Khronos President

Welcome!

- **An exploration of SOC capabilities from the programmer's perspective**
  - How is mobile silicon interfacing to mobile apps?
- **Overview of acceleration APIs on today's mobile OS**
  - And how they can be used to optimize performance and power
- **Focus on mobile innovation hotspots**
  - Vision and gesture processing, Augmented Reality, Sensor Fusion, Computational Photography, 3D Graphics
- **Highlight silicon-level opportunities and challenges still be to solved**
  - While exploring the state of the art in mobile programming

SOC =
'System On Chip'
Minus memory and some peripherals



Speakers

Session	Speaker	Company	Title
Connecting Mobile SOC Hardware to Apps	Neil Trevett	Khronos	Khronos President and NVIDIA VP of Mobile Content
Break		Break	
Camera and Video	Sean Mao	ArcSoft	VP Marketing, Advanced Imaging Technologies
Vision and Gesture Processing	Itay Katz	Eyesight	Co-Founder & CTO
Augmented Reality	Ben Blachnitzky	Metaio	Director of R&D
Sensor Fusion	Jim Steele	Sensor Platforms	VP Engineering
3D Gaming	Daniel Wexler	the11ers	CXO
Panel Session		All Speakers	

Khronos Connects Software to Silicon

- **Khronos APIs define processor acceleration capabilities**
  - Graphics, video, audio, compute, vision and sensor processing

APIs developed today define the functionality of platforms and devices tomorrow

APIs BY the Industry FOR the Industry

- **Khronos defines APIs at the software silicon interface**
  - Low-level "Foundation" functionality needed on every platform
- **Khronos standards have strong industry momentum**
  - 100s of man years invested by industry experts
  - Shipping on billions of devices across multiple operating systems
  - Rigorous conformance tests for cross-vendor consistency
- **Khronos is OPEN for any company to join and participate**
  - Standards are cooperative – one company, one vote
  - Proven legal and IP framework for industry cooperation
  - Khronos membership fees to cover expenses
- **Khronos standards are FREE to use**
  - Members agree to not request royalties

Silicon

Software

HOT CHIPS

KHRONOS GROUP

Over 100 members – any company worldwide is welcome to join

Board of Promoters

API Standards Evolution

DESKTOP

OpenGL

OpenCL

New API technology first evolves on high-end platforms

MOBILE

OpenSL ES

OpenMAX

OpenGL ES

Mobile is the new platform for apps innovation. Mobile APIs unlock hardware and conserve battery life

INTEROP, VISION AND SENSORS

OpenVL

EGL

StreamInput

Apps embrace mobility's unique strengths and need complex, interoperating APIs with rich sensory inputs e.g. Augmented Reality

WEB

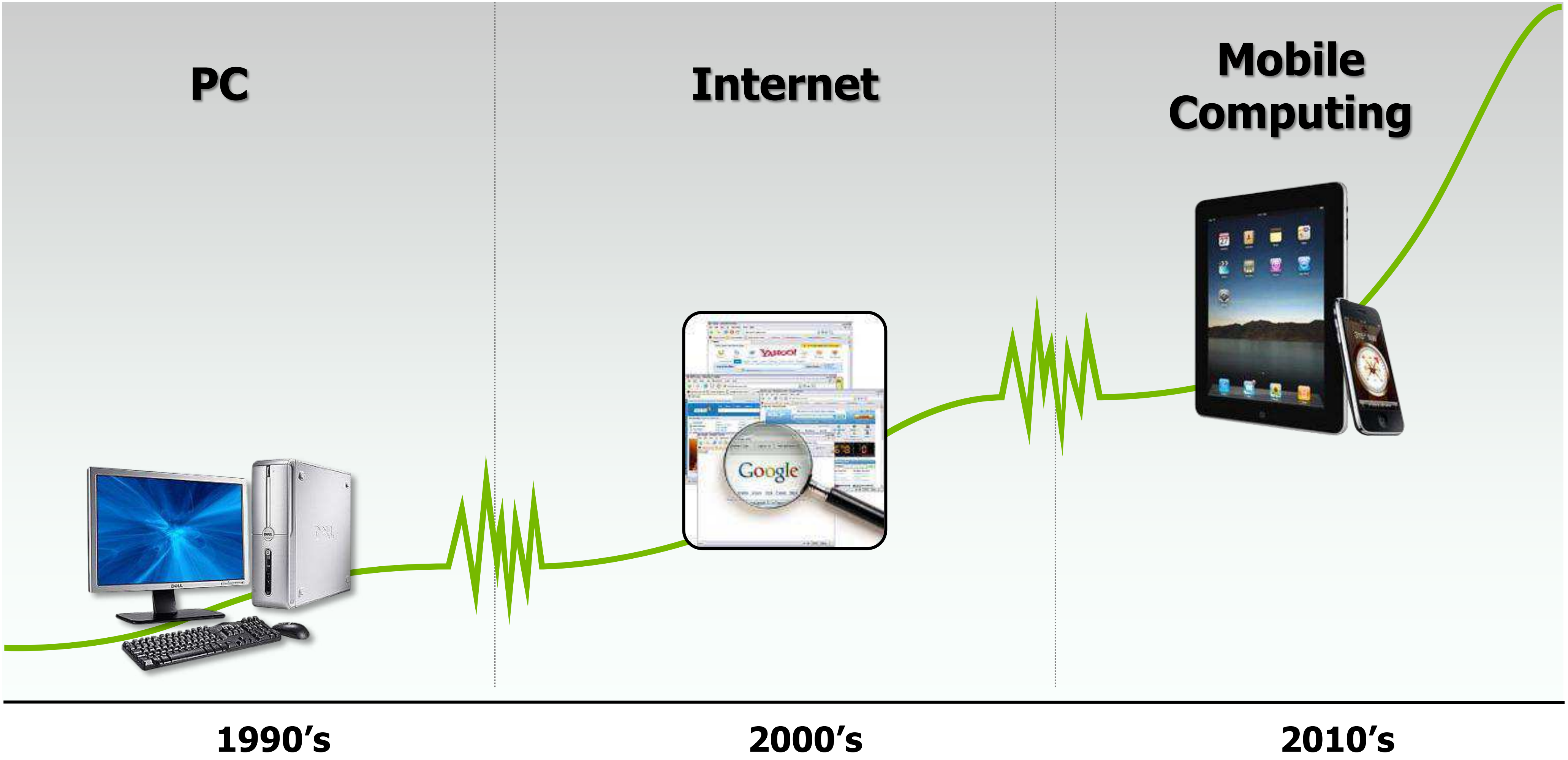
HTML

WebGL

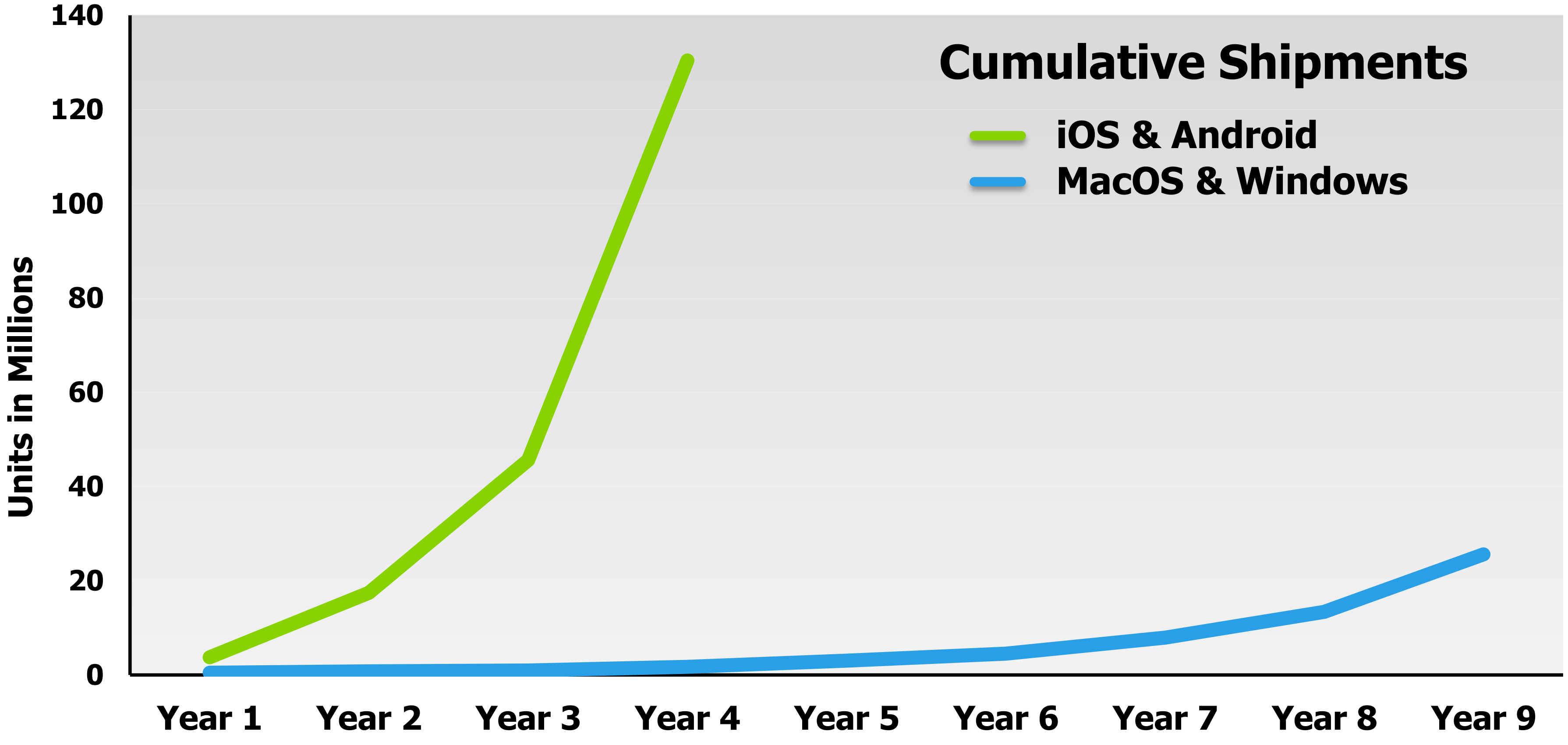
WebCL

Diverse platforms – mobile, TV, embedded – means HTML5 will become increasingly important as a universal app platform

A New Era in Computing



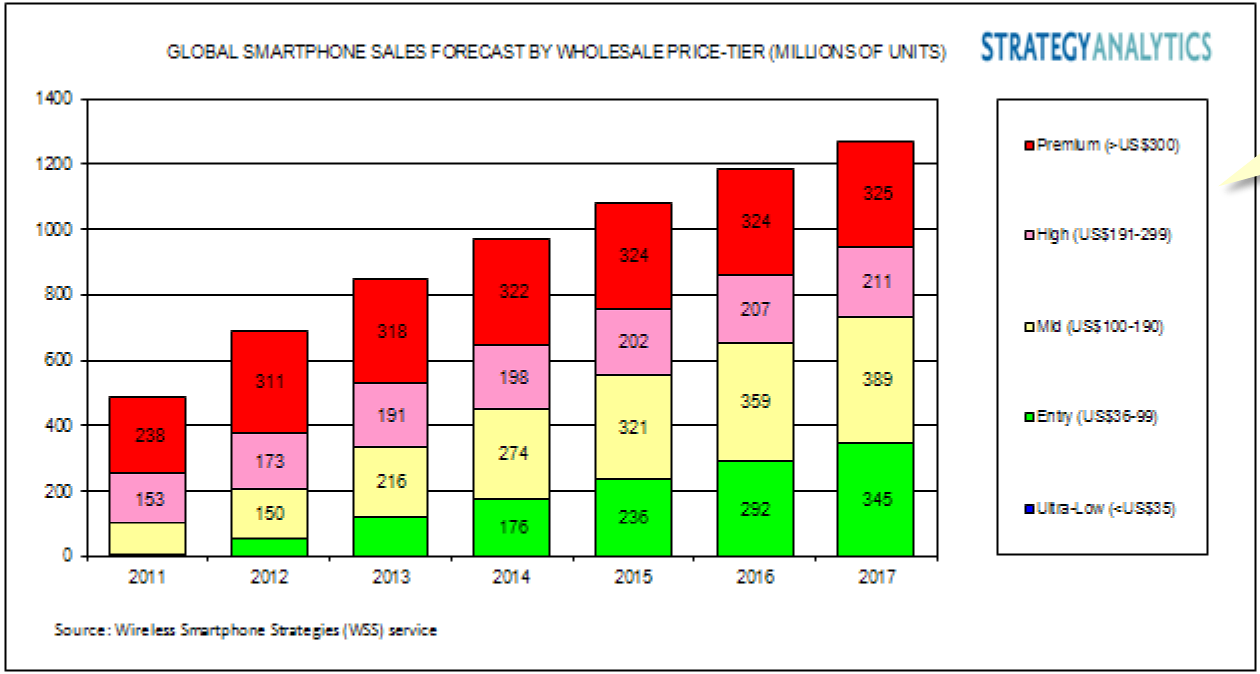
20 Years Faster to 100M Per Year



Source: Gartner, Apple, NVIDIA

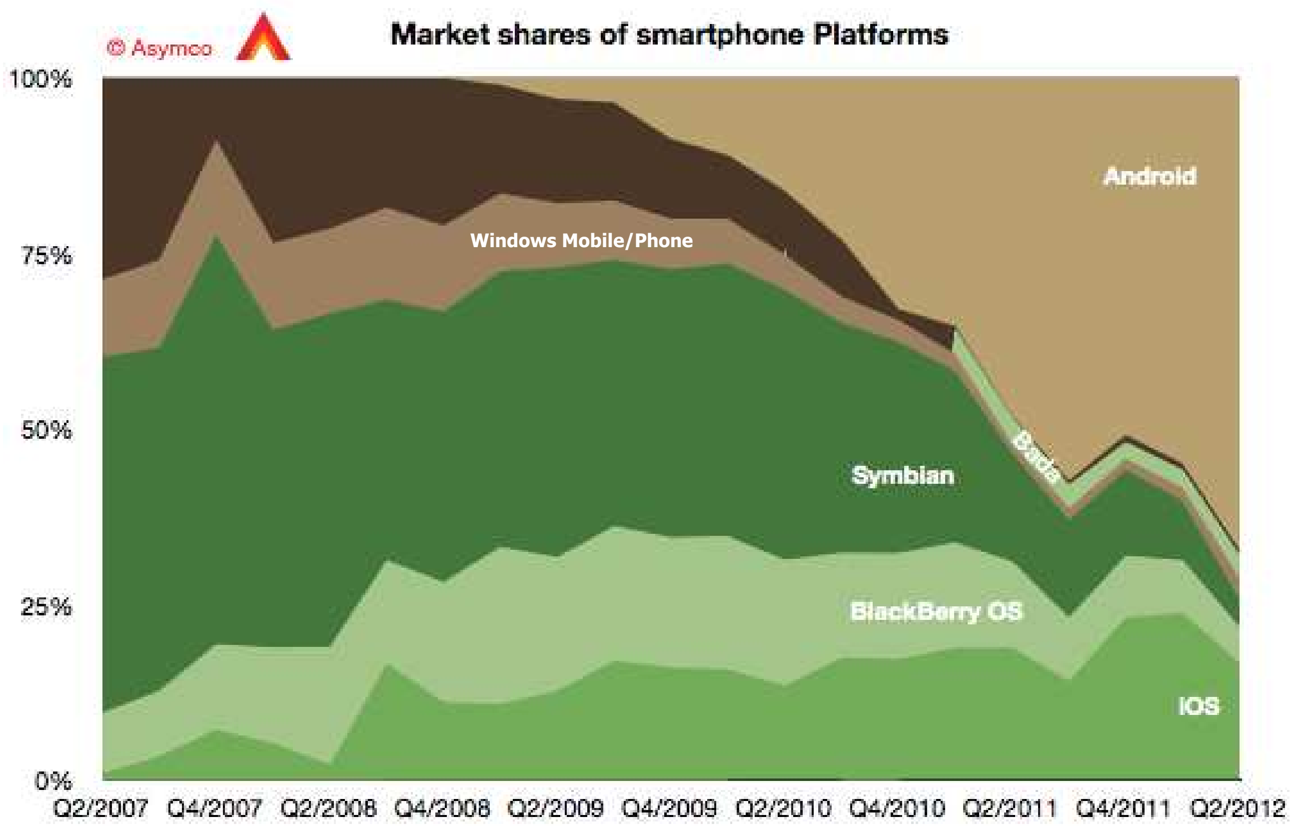
The Largest Device Market Ever

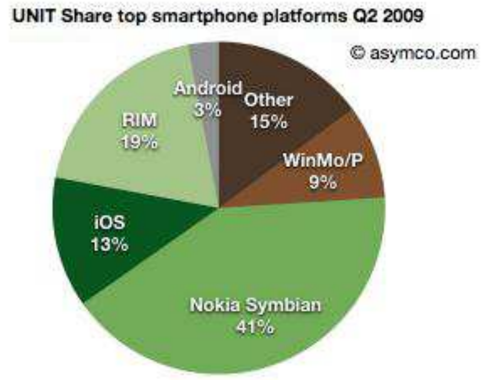
- **IDC - 1.8 billion mobile phones will ship in 2012**
  - By the end of 2016, 2.3 billion mobile phones will ship per year

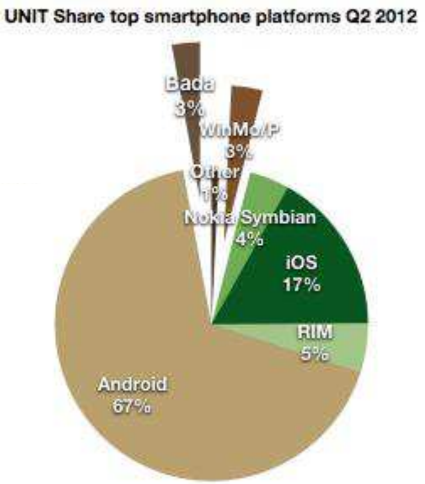


Smart phones account for approximately half of total phone market

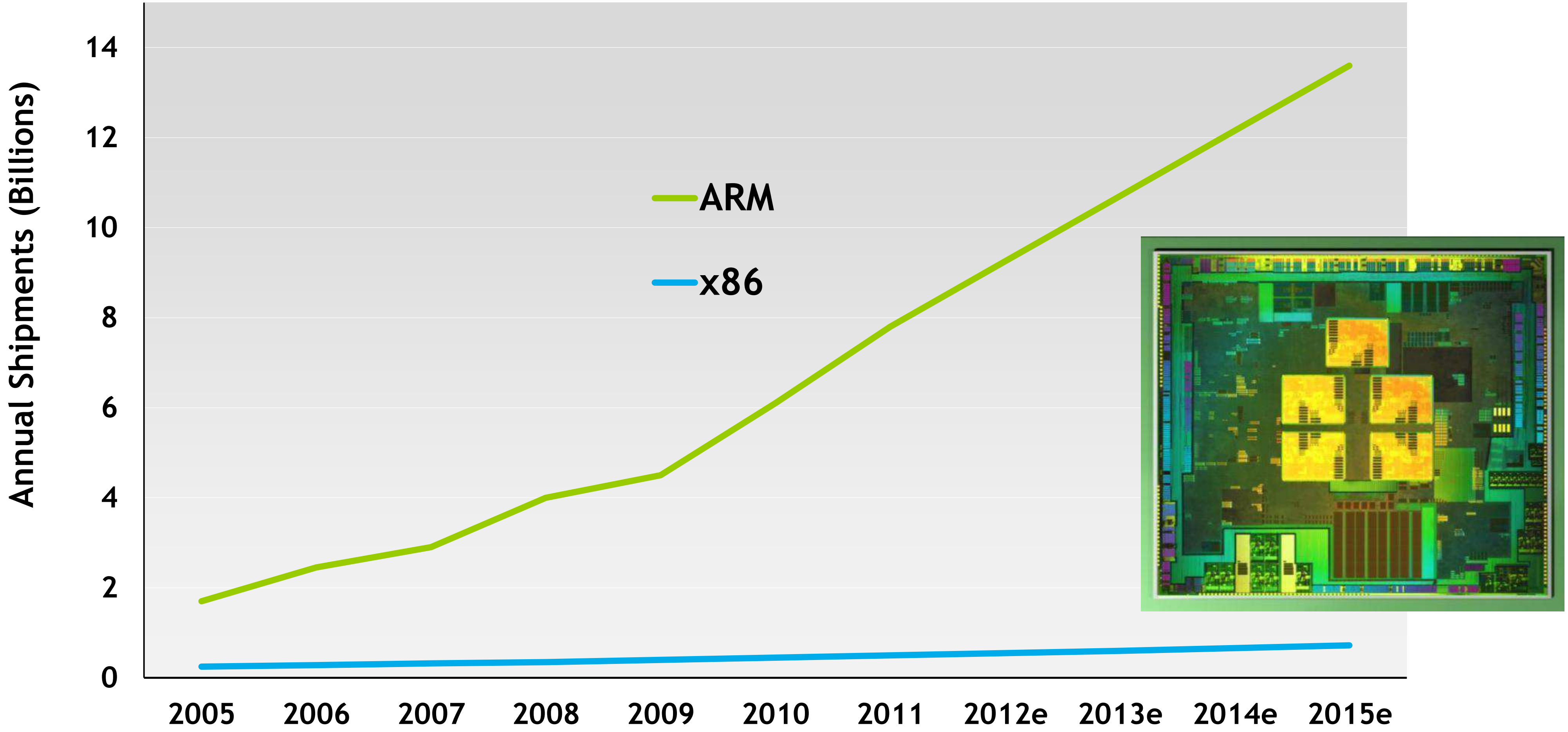
Global Smartphone Market Share





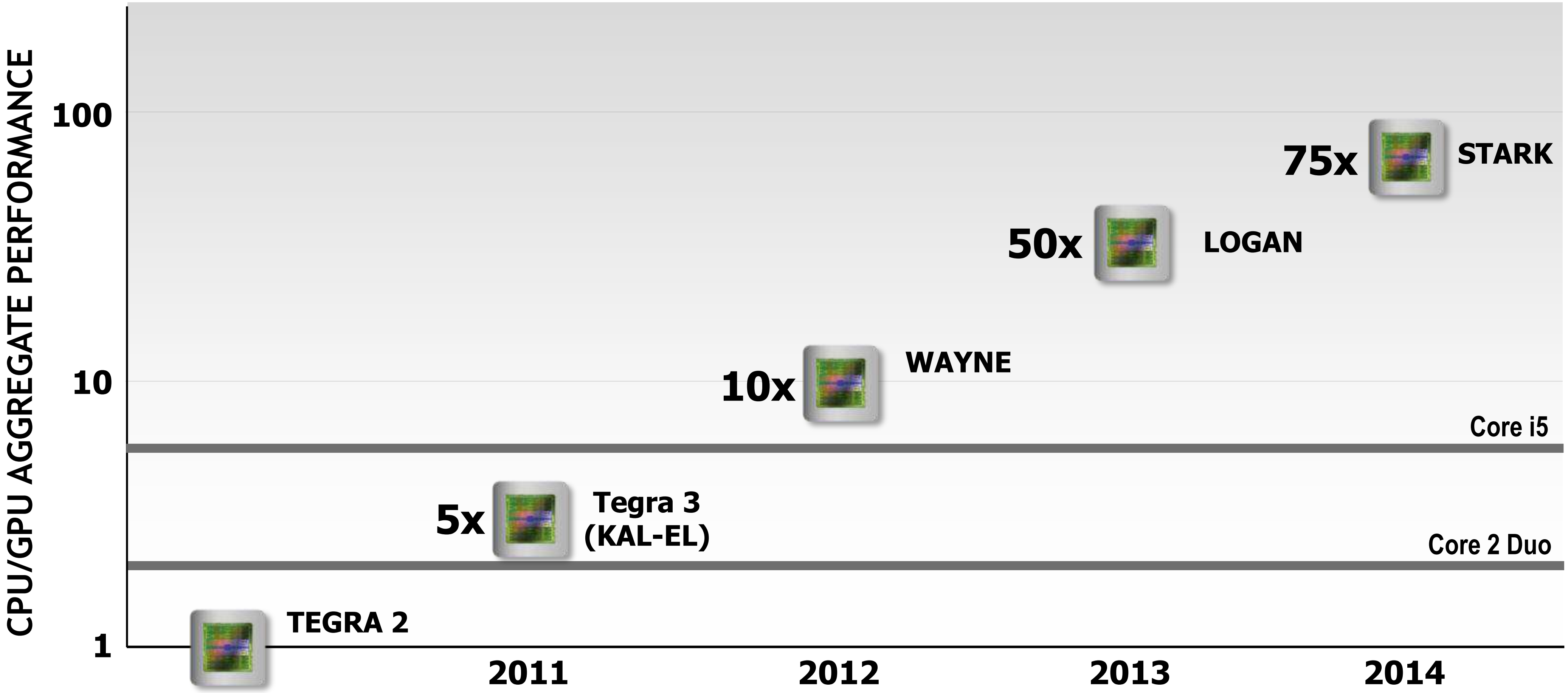


ARM is Licensable and Pervasive



Source: ARM, Mercury Research

Mobile Performance Increases



Power is the New Design Limit

- **The Process Fairy keeps bringing more transistors**
  - Transistors are getting cheaper
- **The End of Voltage Scaling**
  - The Process Fairy isn't helping as much on power as in the past

In the Good Old Days

Leakage was not important, and voltage scaled with feature size

$L' = L/2$

$V' = V/2$

$E' = CV^2 = E/8$

$f' = 2f$

$D' = 1/L^2 = 4D$

$P' = P$

Halve L and get 4x the transistors and 8x the capability for

the same power

The New Reality

Leakage has limited threshold voltage, largely ending voltage scaling

$L' = L/2$

$V' = {\sim}V$

$E' = CV^2 = E/2$

$f' = {\sim}2f$

$D' = 1/L^2 = 4D$

$P' = 4P$

Halve L and get 4x the transistors and 8x the capability for

4x the power!!

Mobile Thermal Design Point

4-5" Screen takes 250-500mW

2-4W

7" Screen takes 1W

4-7W

10" Screen takes 1-2W
Resolution makes a difference!
The iPad3 screen takes up to 8W

6-10W

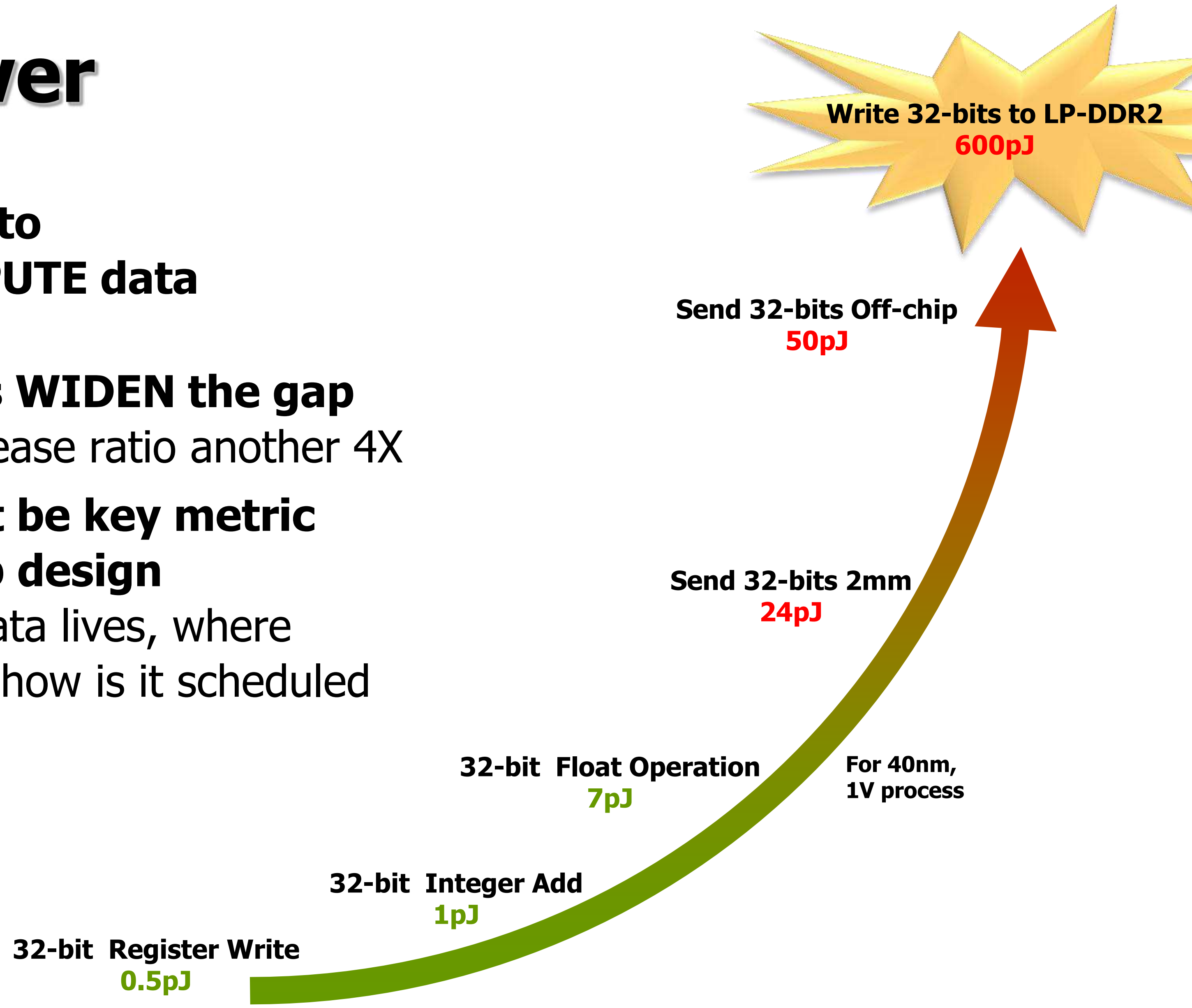
30-90W

Max system power before thermal failure

Even as battery technology improves - these thermal limits remain

Apps and Power

- **Much more expensive to MOVE data than COMPUTE data**
- **Process improvements WIDEN the gap**
  - 10nm process will increase ratio another 4X
- **Energy efficiency must be key metric during silicon AND app design**
  - Awareness of where data lives, where computation happens, how is it scheduled



Energy Optimization Opportunities

- **Dark Silicon**
  - Lots of space for transistors – just can't turn them all on at same time
  - Multiple specialized hardware units that are only turned on when needed
  - Increase locality and parallelism of computation to save power compared to programmable processors
- **Dynamic and feedback-driven software power optimization**
  - Instrumentation for energy-aware compilers and profilers
  - Most compilers just look at one thread, take a more global view
  - Power optimizing compiler back-end / installers
- **Smart, holistic use of sensors and peripherals**
  - Wireless modems and networks
  - Motion sensors, cameras, networking, GPS

Camera Sensor Processing

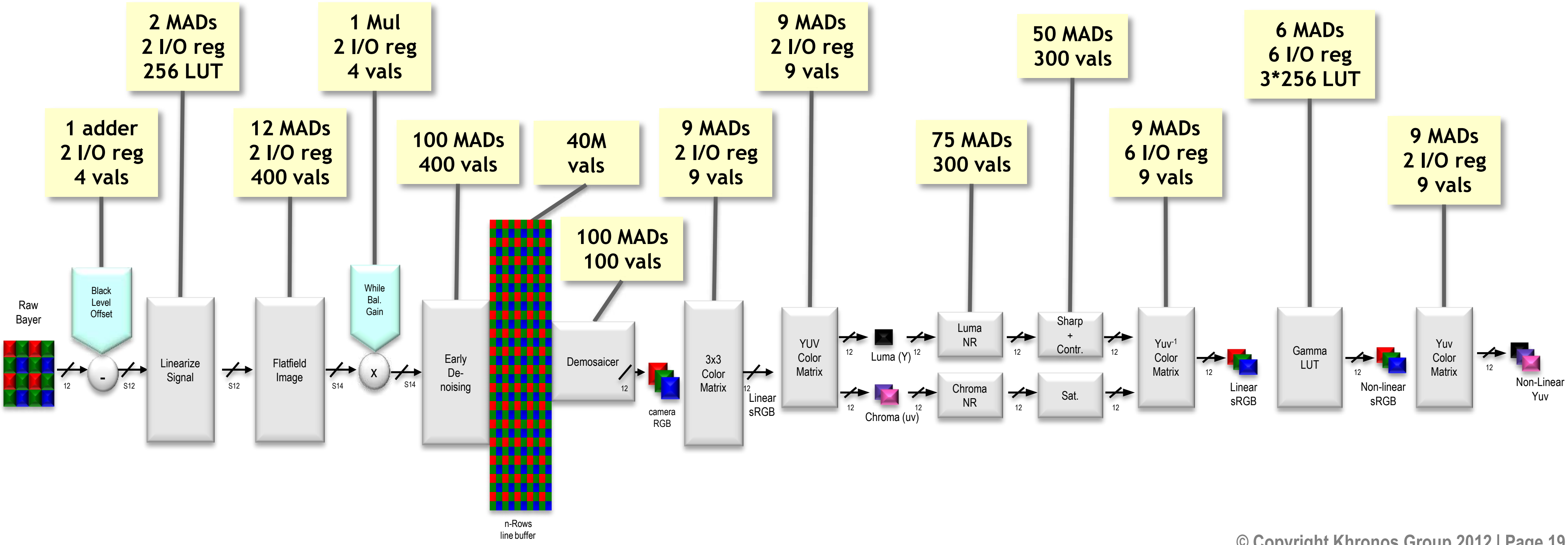
- **CPU**
  - Single processor or Neon SIMD
  - Makes heavy use of general memory
  - Non-optimal performance and power
- **GPU**
  - Many way parallelism
  - Efficient image caching into general memory
  - Programmable and flexible
  - Still significant use of cache/memory
- **Camera ISP = Image Signal Processor**
  - Scan-line-based
  - Data flows through compact hardware pipe
  - No global memory used to minimize power
  - Little or no programmability

Original Scene (shown at 200%) | What Your Camera Sees (through a Bayer array)

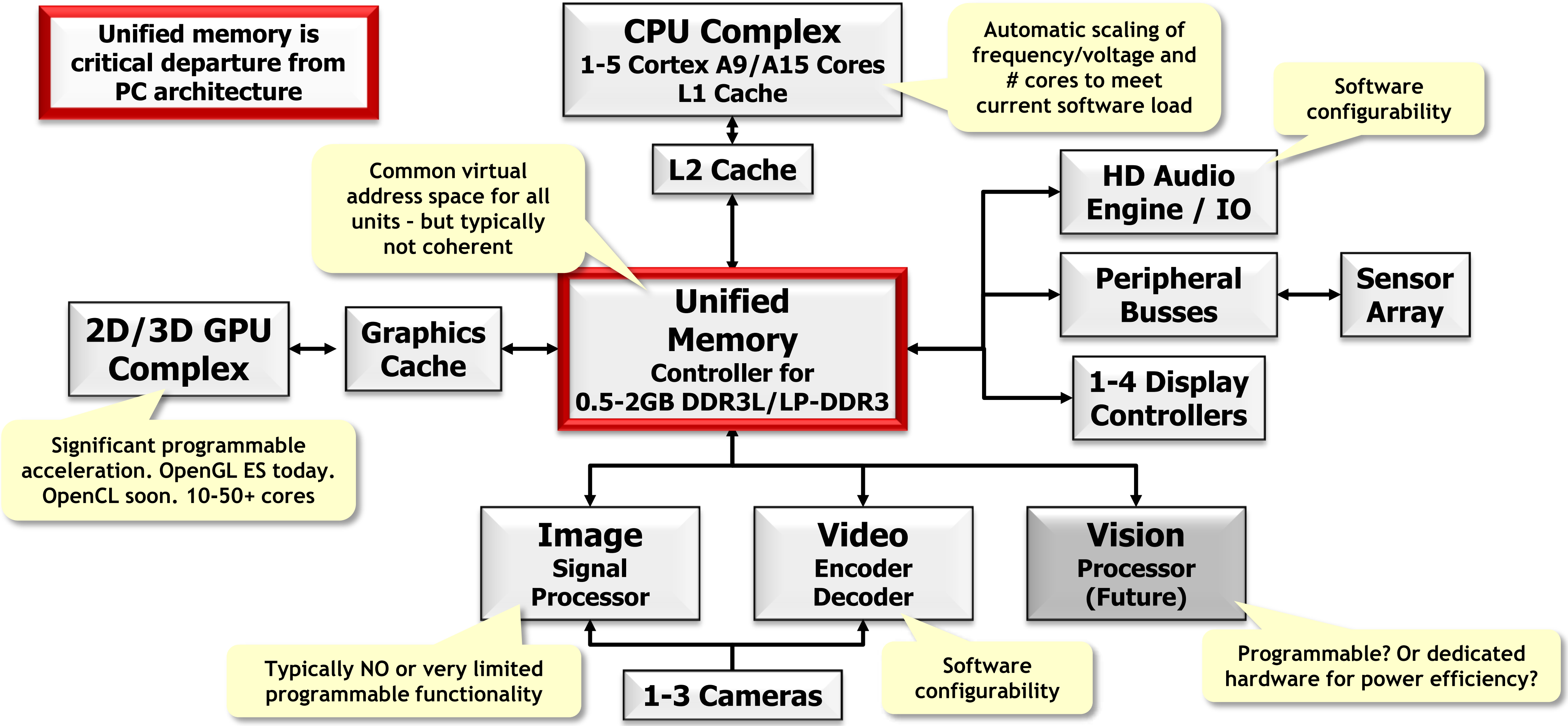
Typical Camera ISP

- ~760 math Ops
- ~42K vals = 670Kb
- 300MHz → ~250Gops

- Computational photography apps beginning to mix non-programmable ISP processing with more flexible GPU or CPU processing
- ISP pipelines could provide tap/insertion points to/from CPU/GPU at critical pipeline points



Programmers View of Typical SOC c. 2012



HSA Feature Roadmap

Heterogeneous System Architecture Foundation: AMD, ARM, Imagination, TI, MediaTek

Physical Integration	Optimized Platforms	Architectural Integration	System Integration
Integrate CPU & GPU in silicon	GPU Compute C++ support	Unified Address Space for CPU and GPU	GPU compute context switch
Unified Memory Controller	User mode scheduling	GPU uses pageable system memory via CPU pointers	GPU graphics pre-emption
Common Manufacturing Technology	Bi-Directional Power Mgmt between CPU and GPU	Fully coherent memory between CPU & GPU	Quality of Service

Time → **Current Innovation**

Mobile Innovation Hot Spots

- New platform capabilities being driven by SILICON and APIs

Console-Class 3D
Performance, Quality, Controllers and TV connectivity

Vision - Camera as sensor
Computational Photography
Gesture Processing
Augmented Reality

OpenMAX

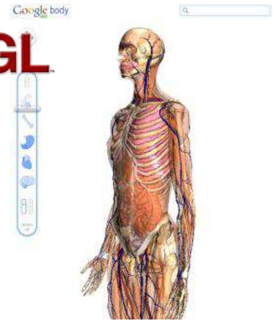
- Media and Image Streaming
- Heterogeneous Parallel Processing

OpenCL

Sensor Fusion
Devices become 'magically' context aware – location, usage, position

StreamInput

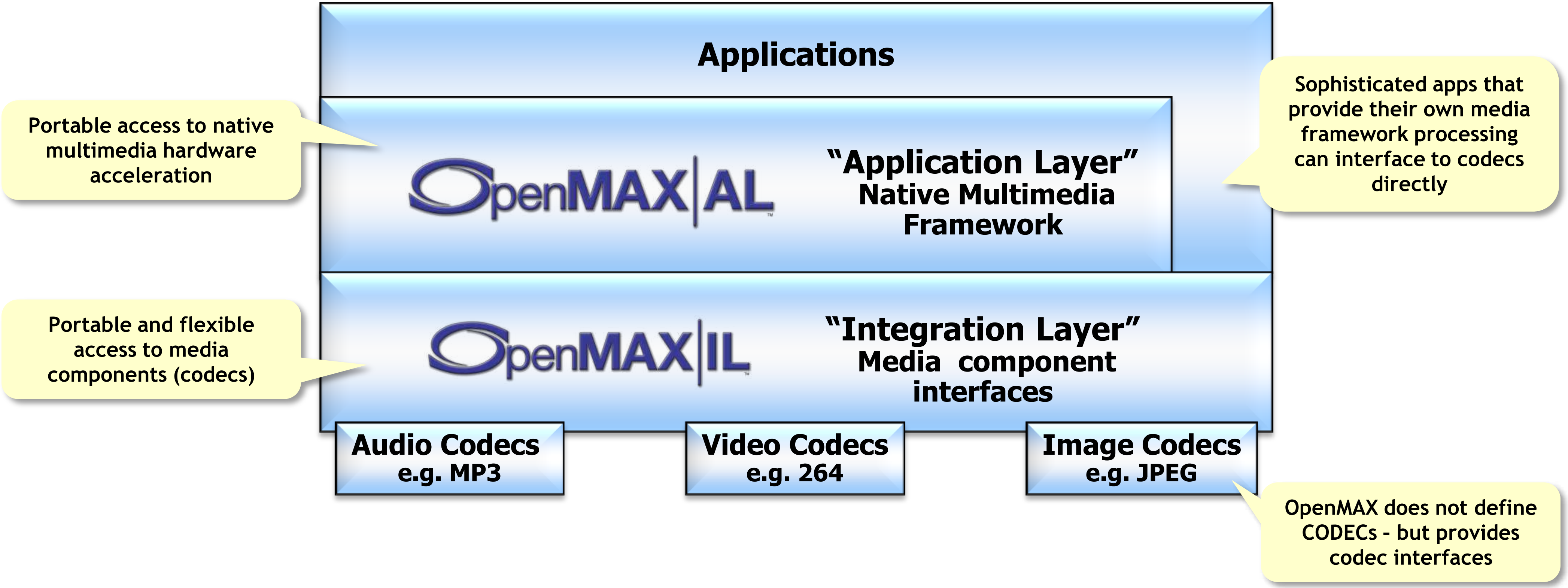
WebGL



HTML5
Web Apps that can be discovered on the Net and run on any platform

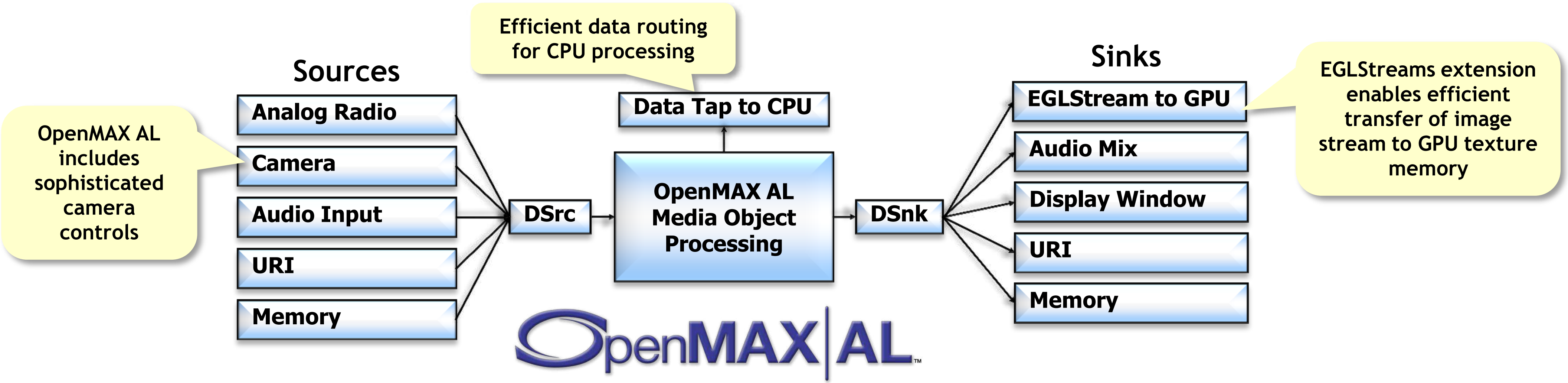
OpenMAX - Media Acceleration

- Family of royalty-free, cross-platform open API standards for video, image stream and camera processing



OpenMAX AL Streaming Media Framework

- **Enables key video, image stream and camera use cases**
  - Enables optimal hardware acceleration with app portability
- **Create Media Objects to play and process images and video with AV sync**
  - Connect to variety of input and output objects to PLAY and RECORD media
- **Full range of video effects and controls**
  - Including playback rate, post processing, and image manipulation



Android Application Development Options

Java apps provide easy access to the Android framework and portability across CPU architectures

".apk" Installable Applications

Dalvik SDK

- Dalvik (Java)
- HTML — WebView Engine

Android NDK

- JNI Java Native Interface to access SDK IO and system resources
- KHRONOS GROUP — Native libraries and APIs
- C/C++

NDK used for performance critical portions of applications such as 3D games, or to access advanced functionality such as extended camera controls

Browser

- HTML
- WebGL Soon!

Applications run in system browser

Apps as web pages - searchable and instantly updated

Accelerating Streaming Media on Android

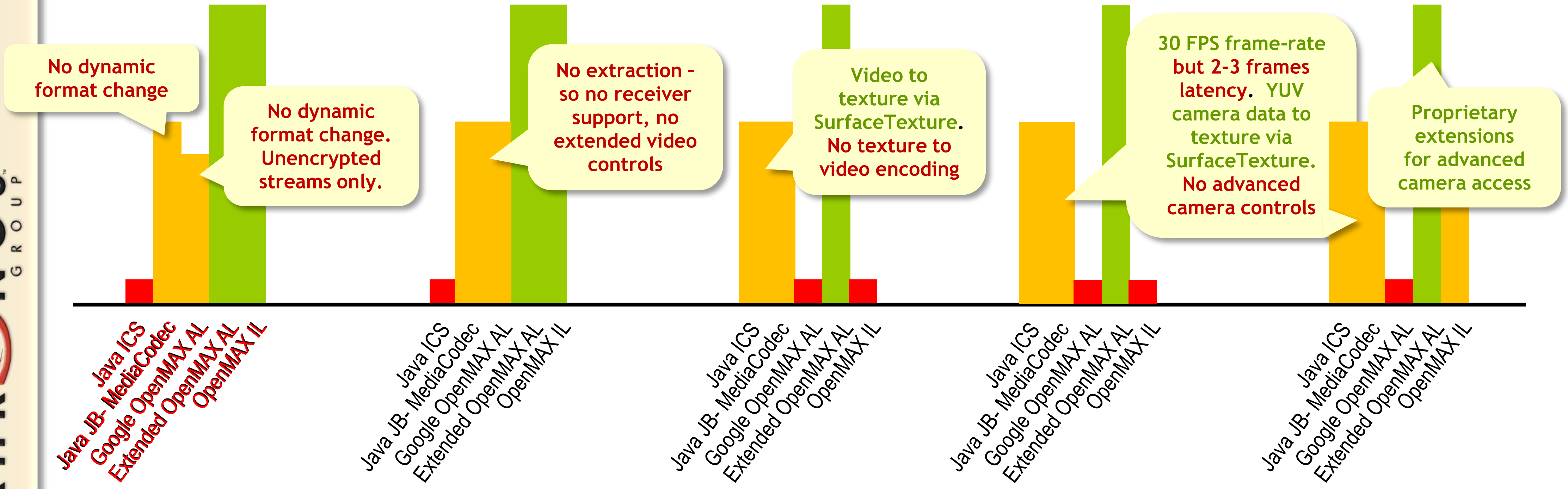
DRM Movies
- Inject encrypted elementary streams into decrypt/decode/render
- Dynamic format changes
- Support for Widevine

HD Video Teleconferencing
- Inject video into decode/render
- Extract video from capture/render
- Extended controls (fine-grained codec query/config, force key frames)

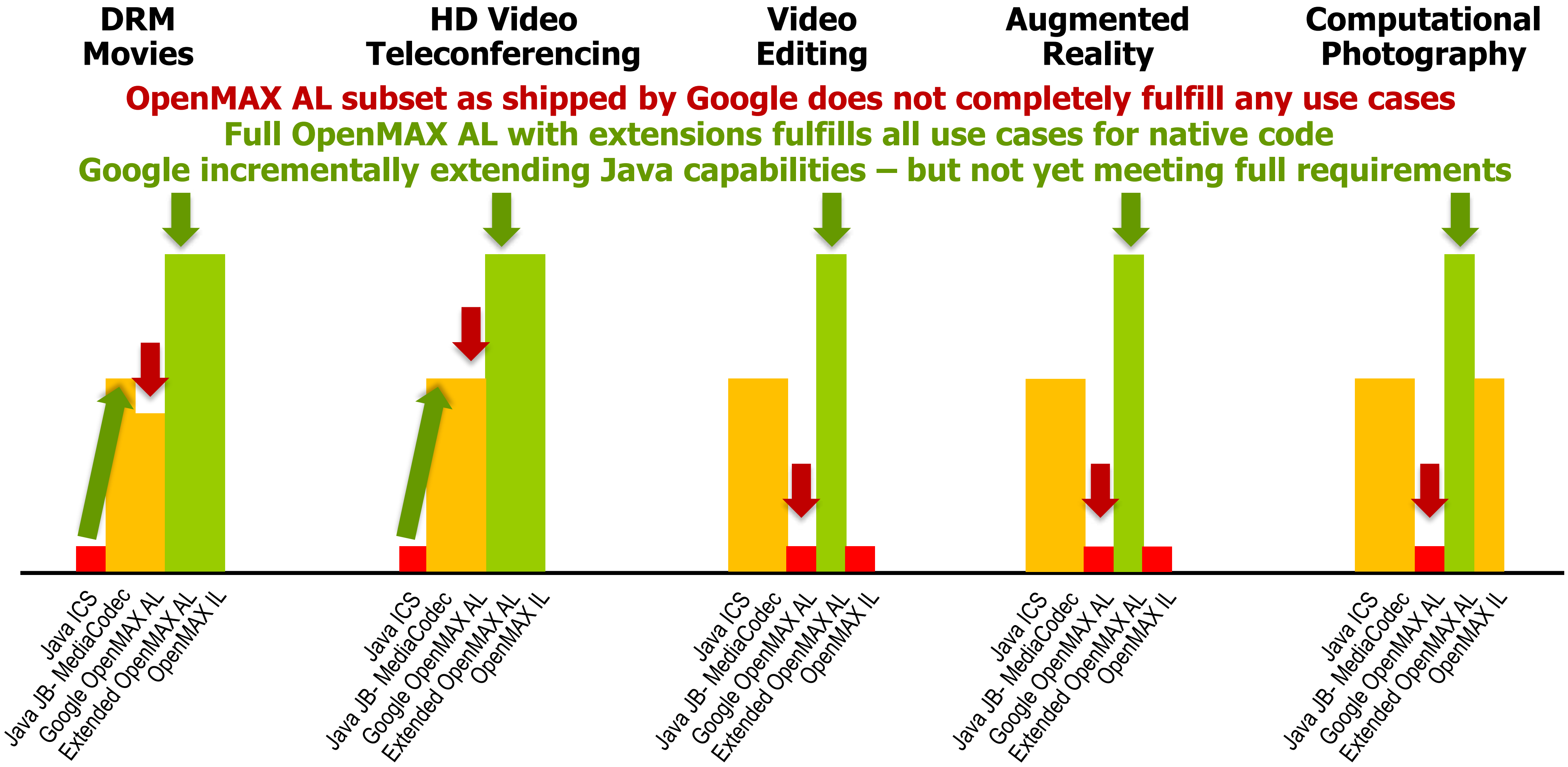
Video Editing
- Video decoding to texture
- Texture to video encoding

Augmented Reality
- High frame-rate, low latency camera capture to app and GPU texture
- Advanced camera control over format, ROI

Computational Photography
- Fast, low latency camera capture to app and GPU texture
- Advanced camera control over format, bracketed burst mode with sequenced key/value pairs



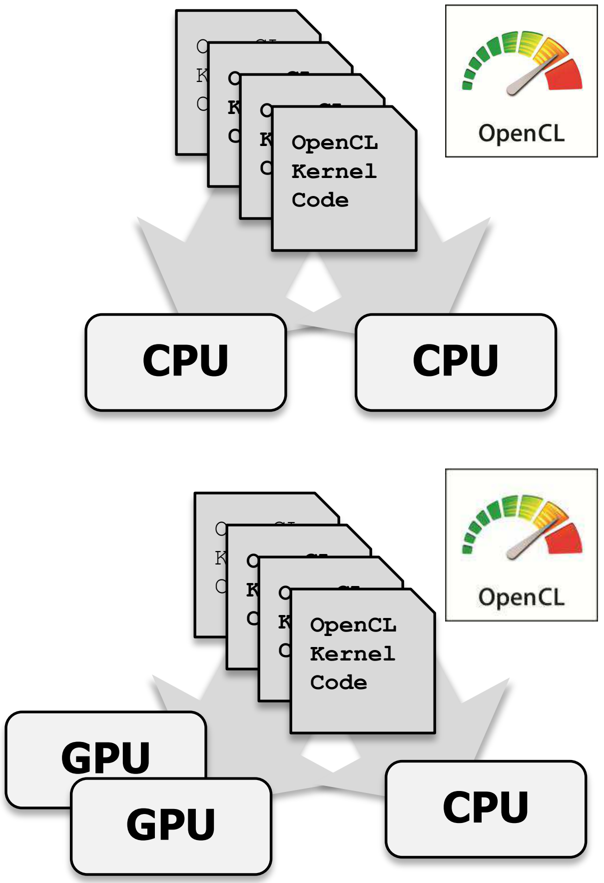
Accelerating Streaming Media on Android



OpenCL – Heterogeneous Computing

- **Native framework for programming diverse parallel computing resources**
  - CPU, GPU, DSP – as well as hardware blocks(!)
- **Powerful, low-level flexibility**
  - Foundational access to compute resources for higher-level engines, frameworks and languages
- **Embedded profile**
  - No need for a separate "ES" spec
  - Reduces precision requirements

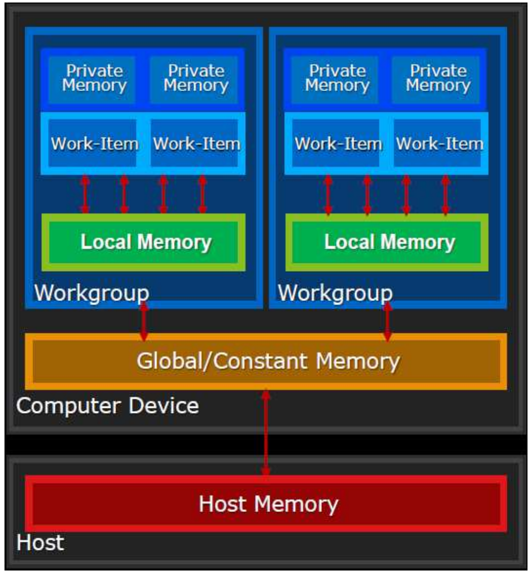
A cross-platform, cross-vendor standard for harnessing all the compute resources in an SOC



One code tree can be executed on CPUs or GPUs

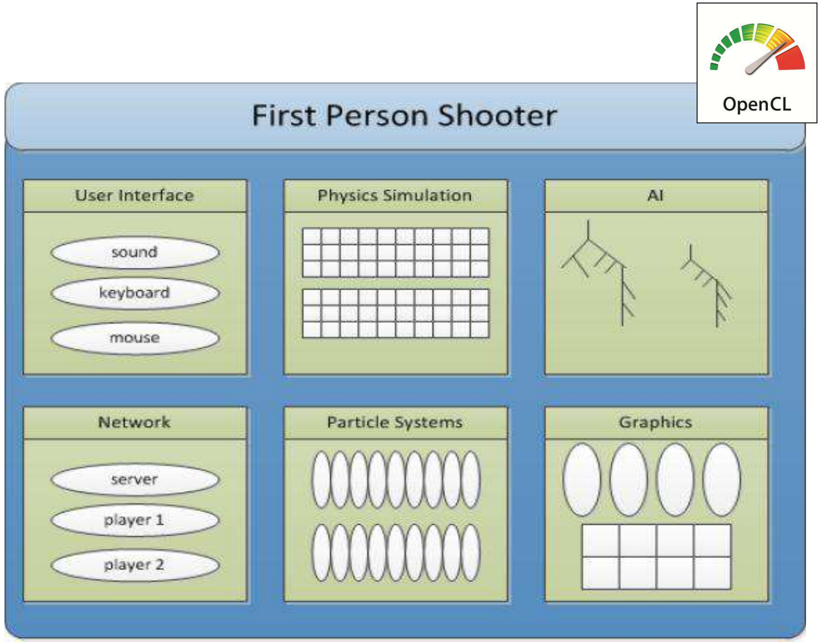
OpenCL Overview

- **C Platform Layer API**
  - Query, select and initialize compute devices
- **Kernel Language Specification**
  - Subset of ISO C99 with language extensions
  - Well-defined numerical accuracy - IEEE 754 rounding with specified max error
  - Rich set of built-in functions: cross, dot, sin, cos, pow, log …
- **C Runtime API**
  - Runtime or build-time compilation of kernels
  - Execute compute kernels across multiple devices
- **Memory management is explicit**
  - Application must move data from host → global → local and back
  - Implementations can optimize data movement in unified memory systems



OpenCL: Execution Model

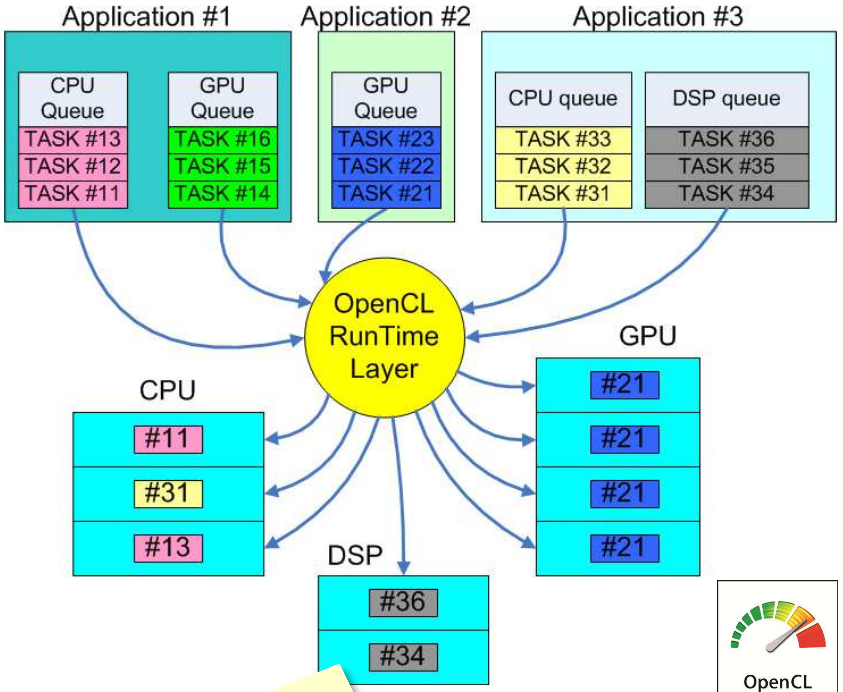
- **Kernel**
  - Basic unit of executable code ~ C function
  - Data-parallel or task-parallel
- **Program**
  - Collection of kernels and functions ~ dynamic library with run-time linking
- **Command Queue**
  - Applications queue kernels & data transfers
  - Performed in-order or out-of-order
- **Work-item**
  - An execution of a kernel by a processing element ~ thread
- **Work-group**
  - A collection of related work-items that execute on a single compute unit ~ core



Example of parallelism types

OpenCL Built-in Kernels

- **Used to control non-OpenCL C-capable resources on an SOC – 'Custom Devices'**
  - E.g. Video encode/decode, Camera ISP …
- **Represent functions of Custom Devices as an OpenCL kernel**
  - Can enqueue Built-in Kernels to Custom Devices alongside standard OpenCL kernels
- **OpenCL run-time a powerful coordinating framework for ALL SOC resources**
  - Programmable *and* custom devices controlled by one run-time



Built-in kernels enable control of specialized processors and hardware from OpenCL run-time

OpenCL Milestones

- **Six months from proposal to released OpenCL 1.0 specification**
  - Due to a strong initial proposal and a shared commercial incentive
- **Multiple conformant implementations shipping on desktop**
  - For CPUs and GPUs on multiple OS
- **18 month cadence between releases**
  - Backwards compatibility protects software investment

Dec08 – OpenCL 1.0 released. Conformance tests released Dec08

Jun10 – OpenCL 1.1 Specification and conformance tests released

Nov11 – OpenCL 1.2 Specification and conformance tests released

2013/4 – OpenCL on mobile platforms begins to ship pervasively

Adobe at SIGGRAPH 2012

Adobe ♥ OpenCL

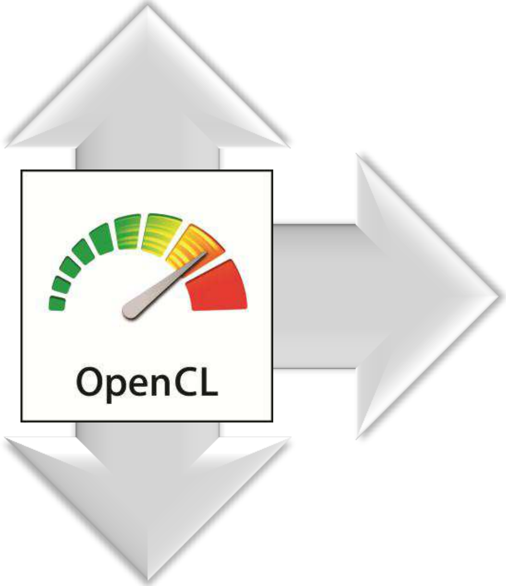
- **Compute API supported across vendors**
- **Programming model familiar to C programmers**
- **Demonstrated performance**
- **Same compute kernels on CPU and GPU!**

- **Adobe is now active member of OpenCL working group**
  - Contributing Adobe's experience and minds to continue OpenCL evolution

SIGGRAPH – Khronos OpenCL BOF - August 8, 2012 Page 7

OpenCL Roadmap

OpenCL-HLM **(High Level Model)**
Exploring high-level programming model, unifying host and device execution environments through language syntax for increased usability and broader optimization opportunities



Discussions include ways to optimize use of unified and shared virtual memory systems

Long-term Core Roadmap
Exploring enhanced memory and execution model flexibility to catalyze and expose emerging hardware capabilities

OpenCL-SPIR **(Standard Parallel Intermediate Representation)**
Exploring LLVM-based, low-level Intermediate Representation for code obfuscation/security and to provide target back-end for alternative high-level languages

OpenCL as Parallel Compute Foundation

HLM	C++ AMP	Aparapi	RenderScript	River Trail	WebCL
C++ syntax/compiler extensions	C++ syntax/compiler extensions	Java language extensions for parallelism	C99 kernels for Dalvik with JIT compilation for device portability	Language extensions to JavaScript	JavaScript binding to OpenCL for initiation of OpenCL C kernels

OpenCL

CUDA or DirectCompute may also be used as compiler targets - but OpenCL provides cross-platform, cross-vendor coverage

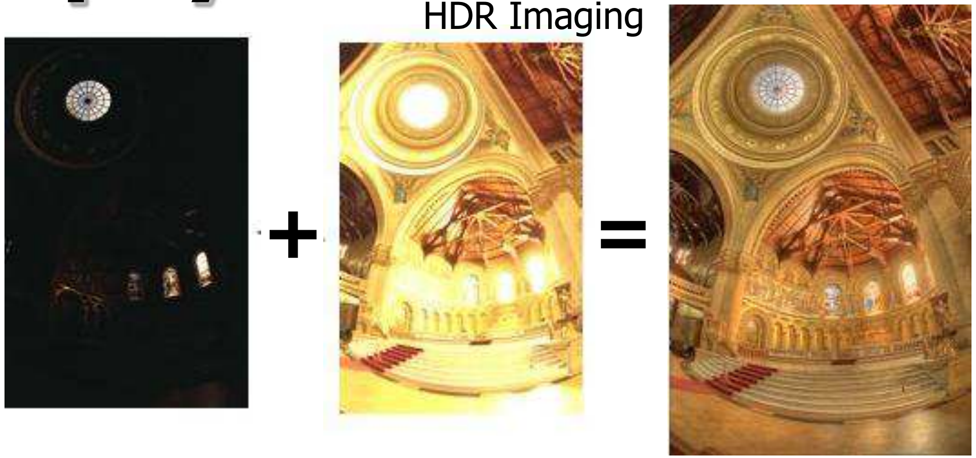
Computational Photography

- **Many advanced photo apps today run on a single CPU**
  - Suboptimal performance and power
- **OpenCL is a platform to harness CPUs/GPUs for advanced imaging**
  - Even if code is 'branchy'

"The tablet … has new multimedia capabilities, including a computational camera, which lets devs tap directly into its computational capability through new application programming interfaces such as OpenCL. That access enables next-generation use cases such as light-field cameras for mobile devices."

QUALCOMM

HDR Imaging



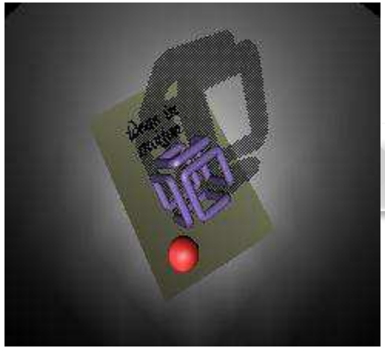
Panorama Stitching

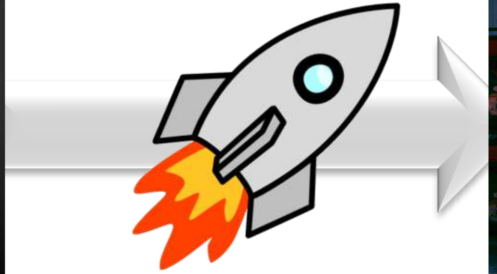
Flash / no-flash imaging

OpenGL 20th Birthday - Then and Now

Launched OpenGL 4.3 at SIGGRAPH 2012

Ideas in Motion - *SGI*

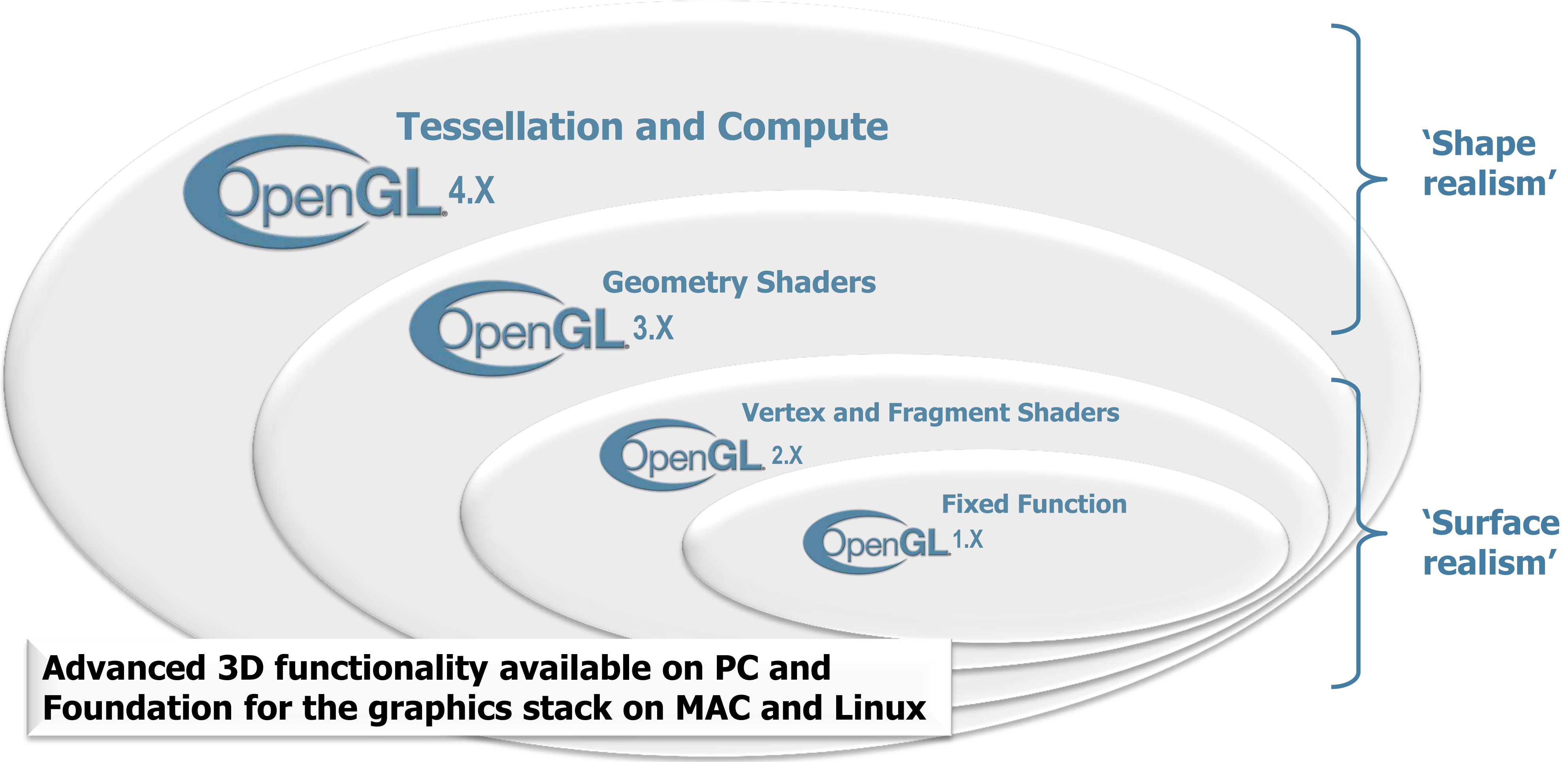




Rage - *id Software*

	1992 Reality Engine 8 Geometry Engines 4 Raster Manager boards	2012 Mobile NVIDIA Tegra 3 Nexus 7 Android Tablet	2012 PC NVIDIA GeForce GTX 680 Kepler GK104
Triangles / sec (millions)	1	103 (x103)	1800 (x1800)
Pixel Fragments / sec (millions)	240	1040 (x4.3)	14,400 (x60)
GigaFLOPS	0.64	15.6 (x25)	3090 (x4830)
	1.5KW	**<5W**	

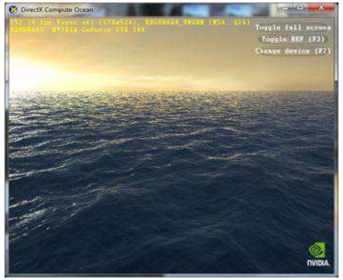
OpenGL for Each Hardware Generation



Advanced 3D functionality available on PC and Foundation for the graphics stack on MAC and Linux

OpenGL 4.3 Compute Shaders

- **Execute algorithmically general-purpose GLSL shaders**
  - Operate on uniforms, images and textures
- **Process graphics data in the context of the graphics pipeline**
  - Easier than interoperating with a compute API IF processing 'close to the pixel'
- **Standard part of all OpenGL 4.3 implementations**
  - Matches DX11 DirectCompute functionality



Physics

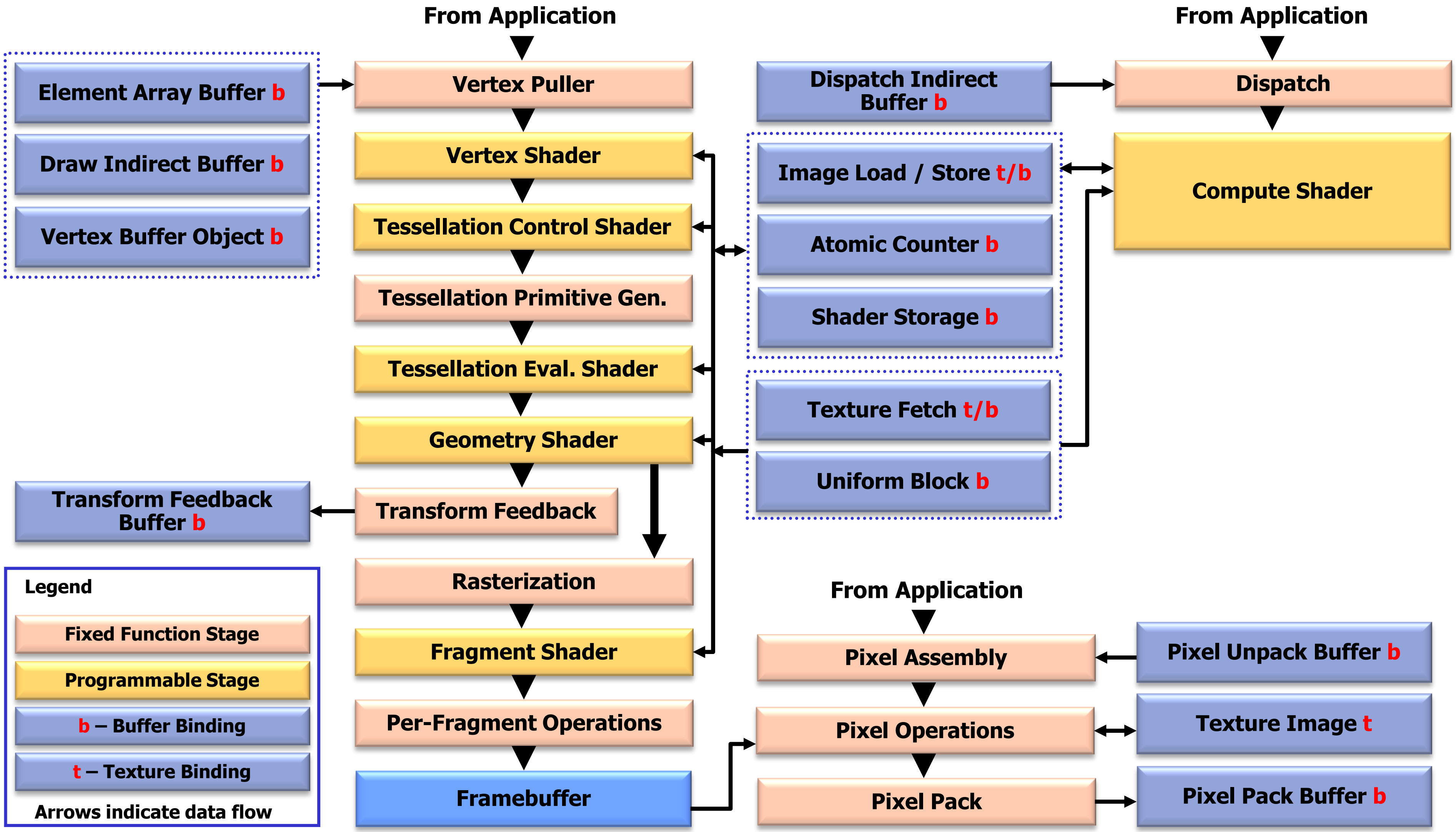
AI Simulation

Ray Tracing

Imaging

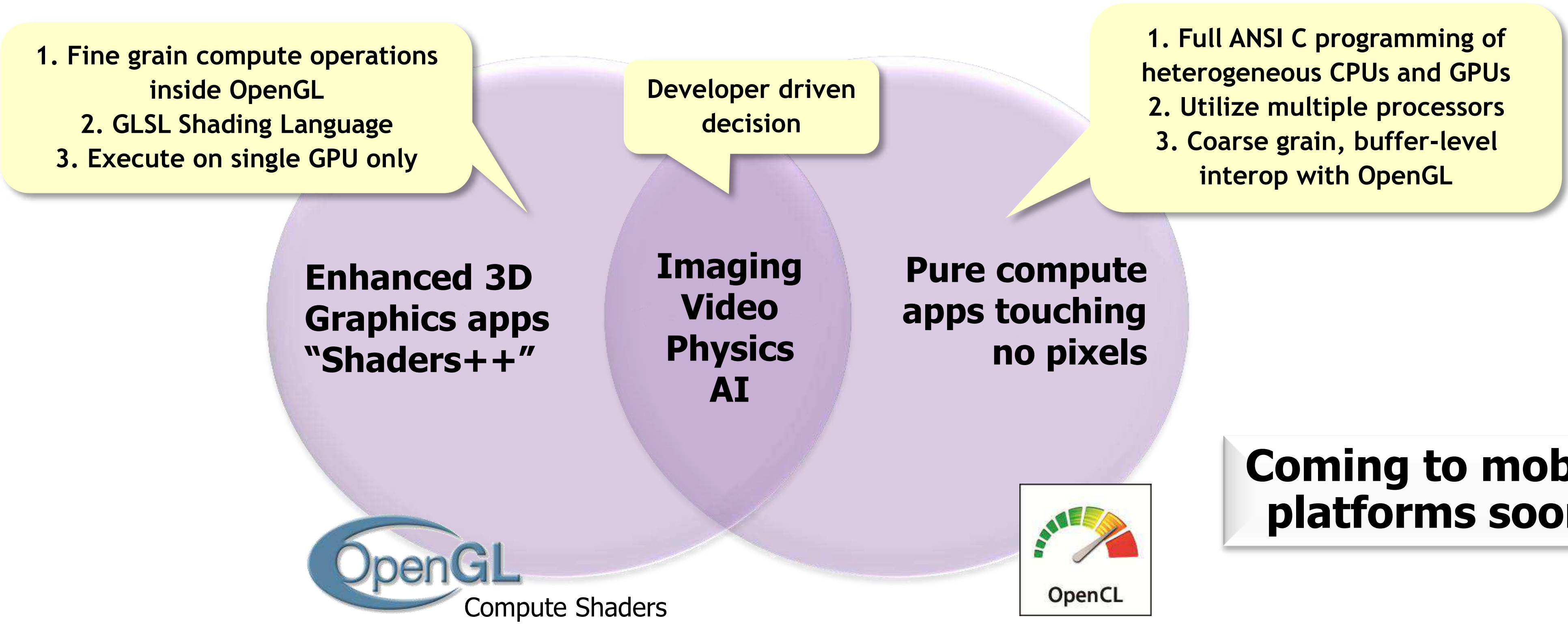
Global Illumination

OpenGL 4.3 with Compute Shaders



OpenCL and OpenGL Compute Shaders

- **OpenGL compute shaders and OpenCL support different use cases**
  - OpenCL provides a significantly more powerful and complete compute solution



Coming to mobile platforms soon!

OpenGL ES

- **Streamlined subset of desktop OpenGL for embedded and mobile devices**

Fixed function 3D Pipeline

OpenGL ES 1.1 Content

Programmable vertex and fragment shaders

OpenGL ES 2.0 Content

ES3 is backward compatible with ES2 so new features can be added incrementally

OpenGL ES 3.0 Content

OpenGL ES 1.1 Platforms Released

OpenGL ES 2.0 Platforms Released

OpenGL ES 3.0 Platforms Released

OpenGL|ES

OpenGL ES 1.0 Spec Released

OpenGL ES 1.1 Spec Released

OpenGL ES 2.0 Spec Released

OpenGL ES 3.0 Spec Released

GL4/DX11-class Capabilities come to ES ??

OpenGL1.5 OpenGL2.0 OpenGL2.1 OpenGL3.0 OpenGL3.1 OpenGL3.2 OpenGL3.3 OpenGL4.0 OpenGL4.1 OpenGL4.2 OpenGL 4.3

OpenGL

OpenGL 4.3 is a superset of DX11

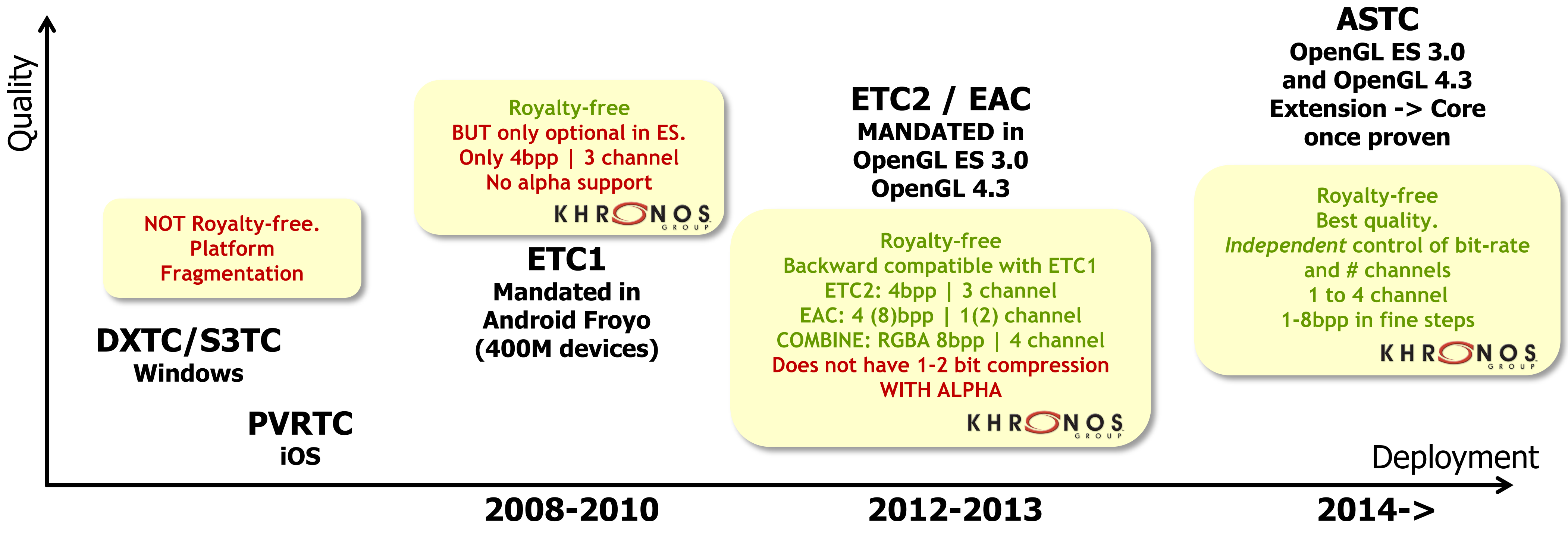
2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015

OpenGL ES 3.0 Highlights

- **Better looking, faster performing games and apps – at lower power**
  - Incorporates proven features from OpenGL 3.3 / 4.x
  - 32-bit integers and floats in shader programs
  - NPOT, 3D textures, depth textures, texture arrays
  - Multiple Render Targets for deferred rendering, Occlusion Queries
  - Instanced Rendering, Transform Feedback …
- **Make life better for the programmer**
  - Tighter requirements for supported features to reduce implementation variability
- **Backward compatible with OpenGL ES 2.0**
  - OpenGL ES 2.0 apps continue to run unmodified
- **Standardized Texture Compression**
  - #1 developer request!

Texture Compression is Key

- **Texture compression saves precious resources**
  - Saves network bandwidth, device memory space AND memory bandwidth
- **Developers need the same texture compression EVERYWHERE**
  - Otherwise portable apps – such as WebGL need multiple copies of same texture



ASTC – Future Universal Texture Standard?

- **Adaptive Scalable Texture Compression (ASTC)**
  - Quality significantly exceeds S3TC or PVRTC at same bit rate
- **Industry-leading orthogonal compression rate and format flexibility**
  - 1 to 4 color components: R / RG / RGB / RGBA
  - Choice of bit rate: from 8bpp to <1bpp in fine steps
- **ASTC is royalty-free and so is available to be universally adopted**
  - Shipping as GL/ES extension today for industry feedback

Original 24bpp

ASTC Compression

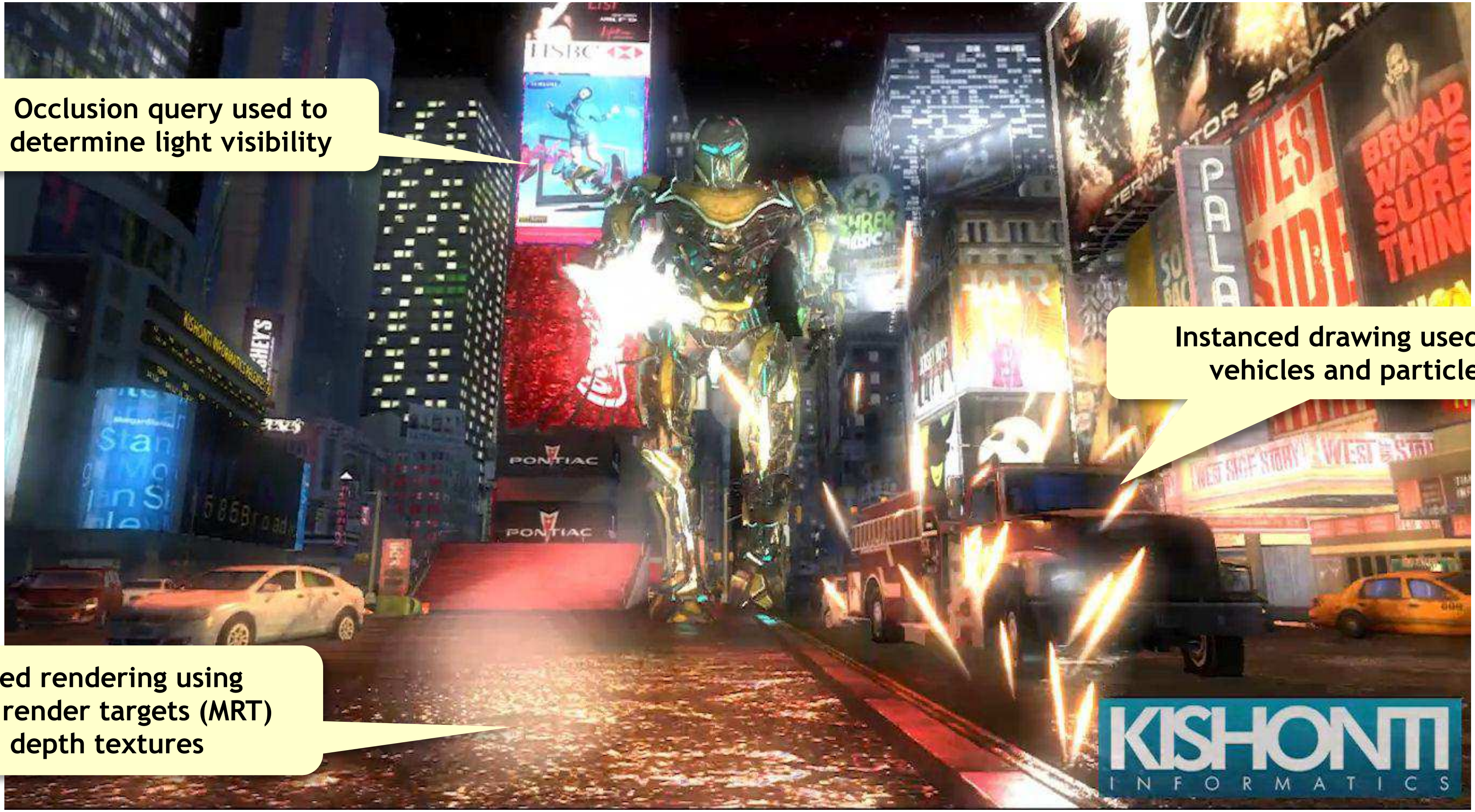
8bpp | 3.56bpp | 2bpp

Kishonti GLBenchmark 3.0

Occlusion query used to determine light visibility

Instanced drawing used for vehicles and particles

Deferred rendering using multiple render targets (MRT) and depth textures



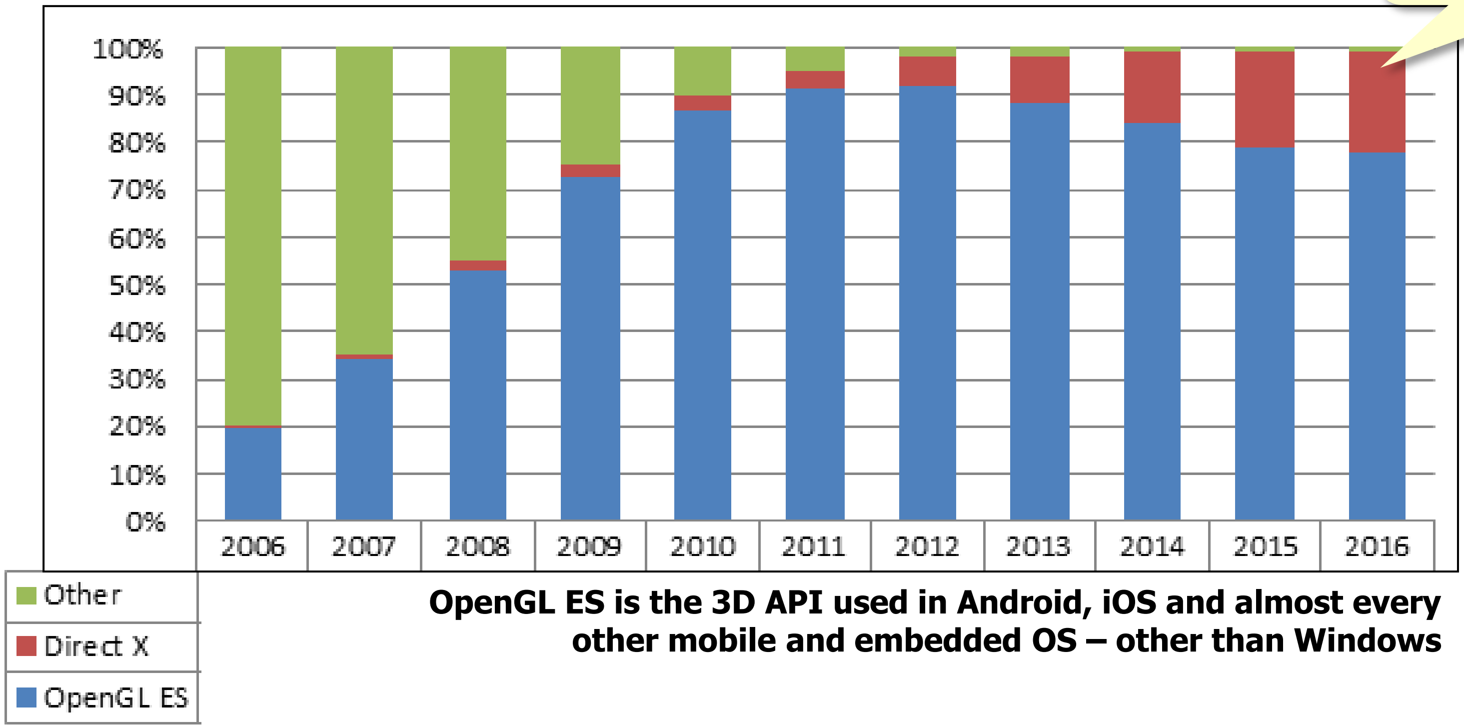
Kishonti "GLBenchmark 3.0" preliminary

OpenGL ES Deployment in Mobile

Use of 3D APIs in Mobile Devices

Source: Jon Peddie Research

On PC – DirectX is used for most apps. On mobile the situation is reversed



OpenGL ES is the 3D API used in Android, iOS and almost every other mobile and embedded OS – other than Windows

Visual-based Augmented Reality

Camera video stream
sent to the compositor

Camera images used to track the camera's location and orientation

3D augmentations composited with video stream

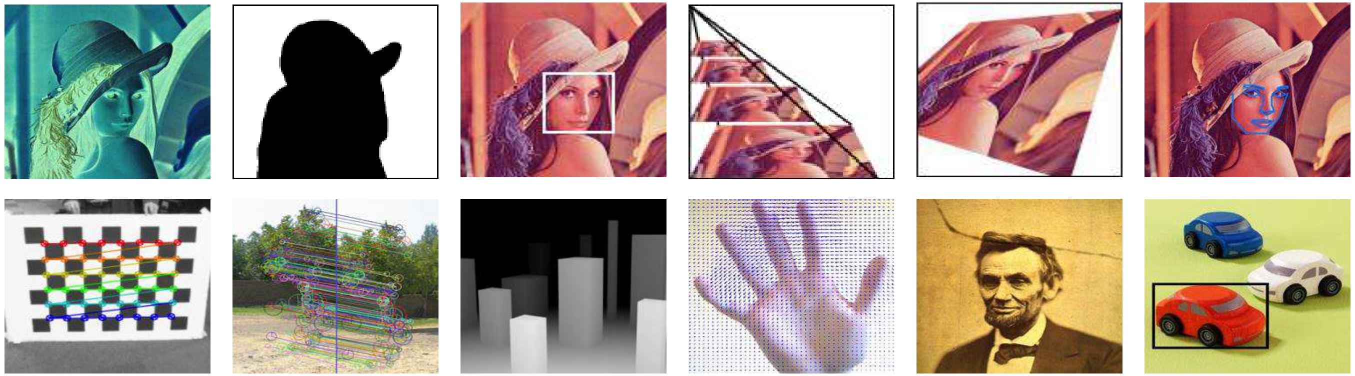
Camera Tracking

3D Augmentation Rendering

Other on-device Sensors

OpenCV

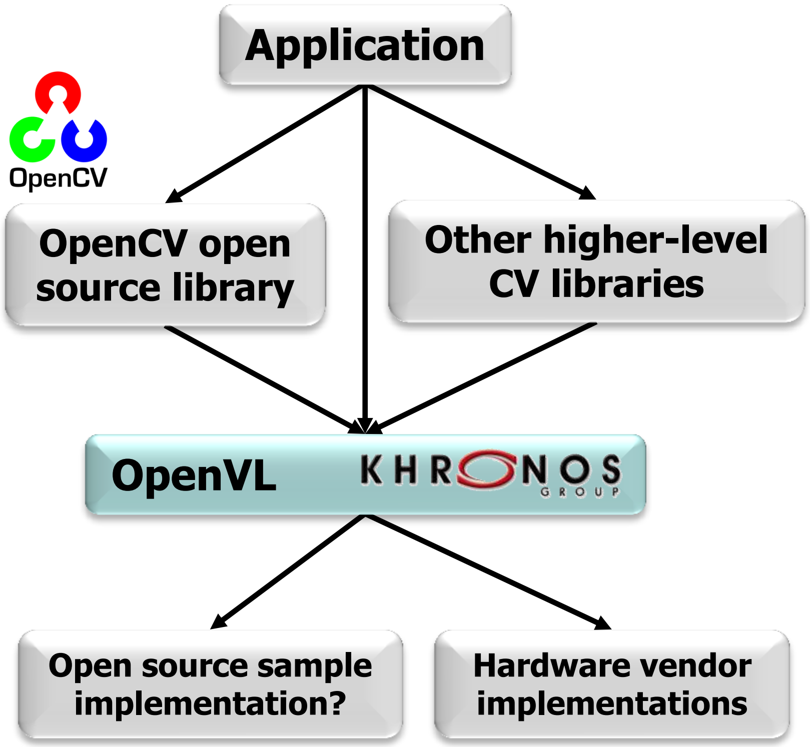
- **Computer vision open source project**
  - Excellent functionality - widely used in academia, fast prototyping, some products
  - Not an API definition and not managed by Khronos
- **Extensive functionality >1,000 functions**
  - Difficult for silicon vendors to provide complete acceleration
- **Traditionally runs on a single CPU**
  - Some partial acceleration projects underway: OpenCL, CUDA, Neon …
  - E.g. MulticoreWare open source CPU/GPU enabled OpenCV over OpenCL



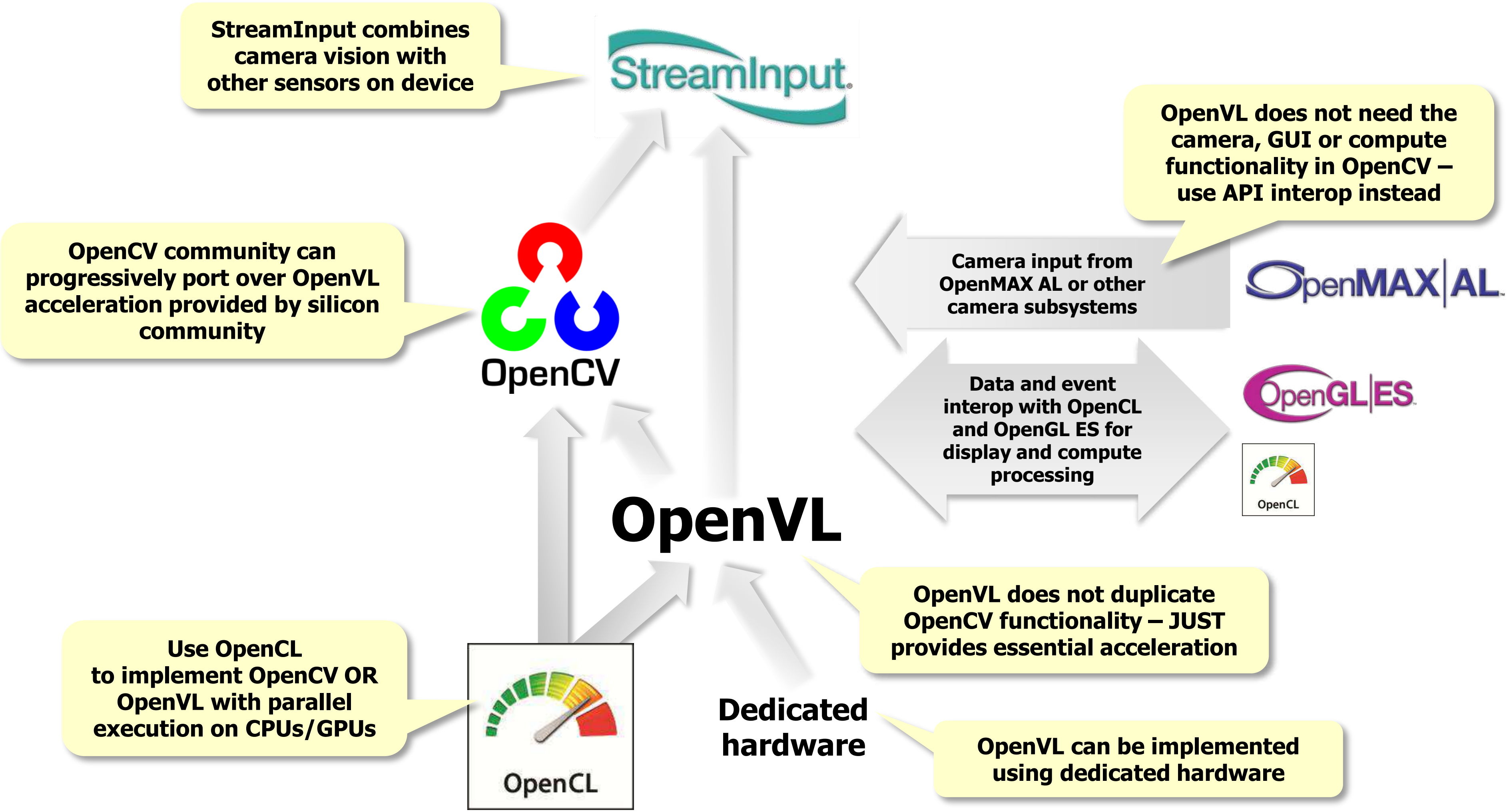
OpenVL

- **Vision Hardware Acceleration Layer**
  - Enable hardware vendors to implement accelerated imaging and vision algorithms
  - For use by high-level libraries or apps directly
- **Primary focus on enabling real-time vision**
  - On mobile and embedded systems
- **Diversity of efficient implementations**
  - From programmable processors to dedicated hardware pipelines

Dedicated hardware can help make vision processing performant and low-power enough for pervasive 'always-on' use



Possible Implementations of Vision Stack

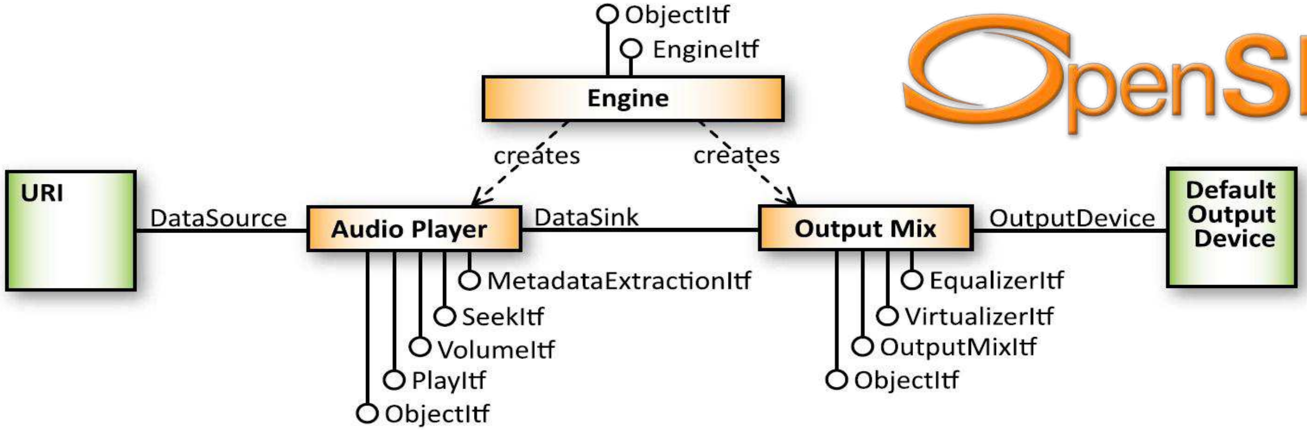


Developer Requested Camera Extensions

- **Query camera information**
  - Focal length (fx, fy), principal point (cx, cy), skew (s), image resolution (h, w)
  - Spatial information of how cameras and sensors are placed on device
  - Calibration and lens distortion
- **FCAM++ for extensive exposure parameters in single or burst mode**
  - Shutter, aperture, ISO, white balance, frame rate, focus modes, resolution
  - Preload parameters for each shot in a burst
- **ROI extraction**
  - From wide angle and fish-eye lenses
- **Data output format control**
  - Grayscale, RGB(A), YUV
  - Access to the raw data e.g. Bayer pattern

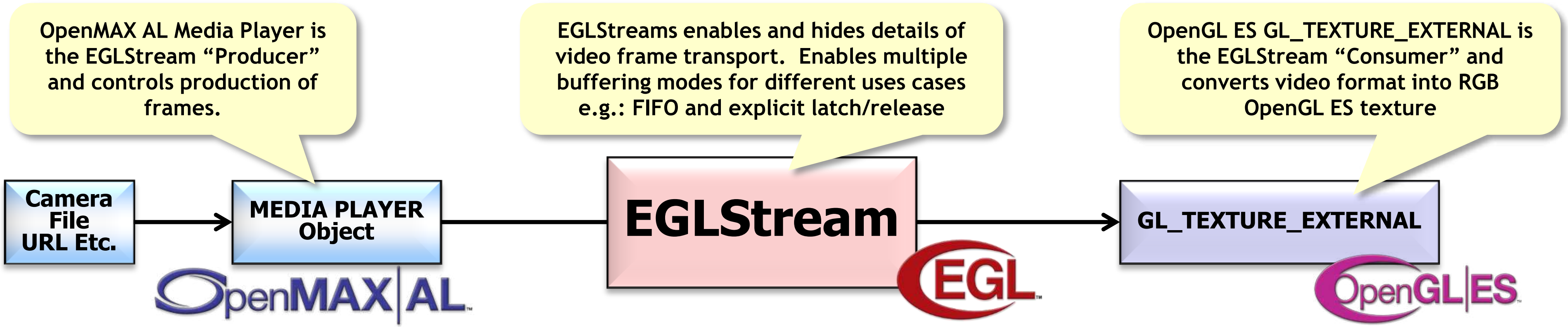
OpenSL ES – Advanced Audio

- **OpenSL ES does for audio what OpenGL ES does for graphics**
  - Advanced audio functionality from simple playback to full 3D positional audio
- **Object-based native audio API for simplicity and high performance**
  - Same object framework as OpenMAX AL
  - Reduces development time
- **Attractive alternative to open source frameworks**
  - Tightly defined specification with full conformance tests
  - Robust application portability across platforms and OS



EGLStream – Streaming Images

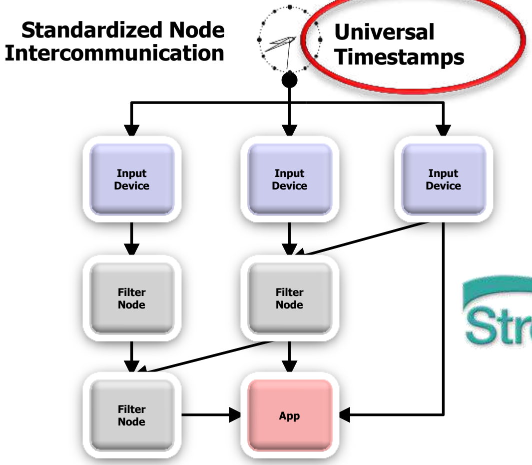
- **EGL – originally embedded version of WGL**
  - Abstraction layer to window systems and memory buffers
- **Role has expanded to provide API interop – data and events**
  - EGLImages – single buffers that can be passed between APIs
  - EGLStreams – provides stream of images – with AV sync
  - Cross process EGLStreams – Producer and Consumer can be in different processes for performance or security – e.g. browser compositing process
- **Android SurfaceTexture is a Java wrapper around EGLStreams**
  - Captures video decode or camera preview to OpenGL ES texture



Portable Access to Sensor Fusion

Apps request semantic sensor information
StreamInput defines possible requests, e.g.
"Provide Skeleton Position" "Am I in an elevator?"

Advanced Sensors Everywhere
RGB and depth cameras, multi-axis motion/position, touch and gestures, microphones, wireless controllers, haptics keyboards, mice, track pads



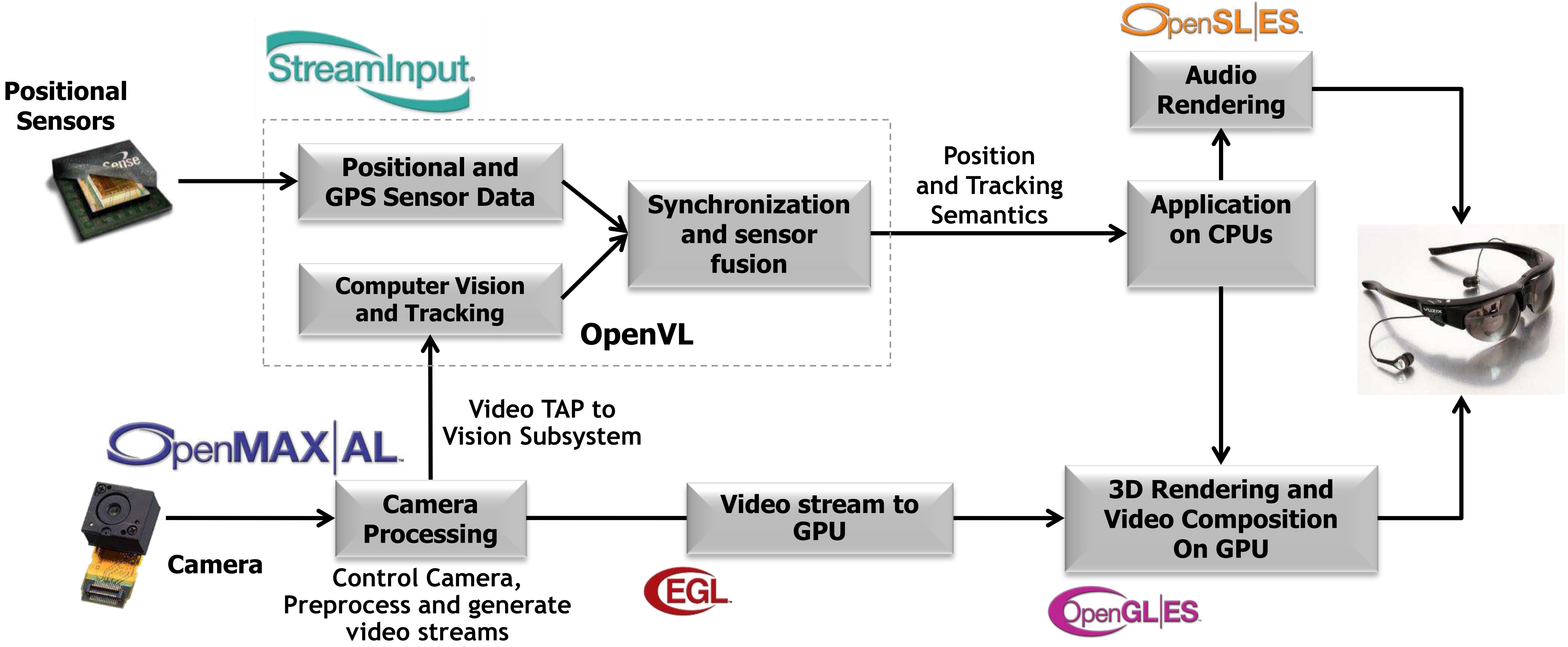
E.g. align samples from camera and other sensors

StreamInput

Apps Need Sophisticated Access to Sensor Data
Without coding to specific sensor hardware

Processing graph provides sensor data stream
Utilizes optimized, smart, sensor middleware
Apps can gain 'magical' situational awareness

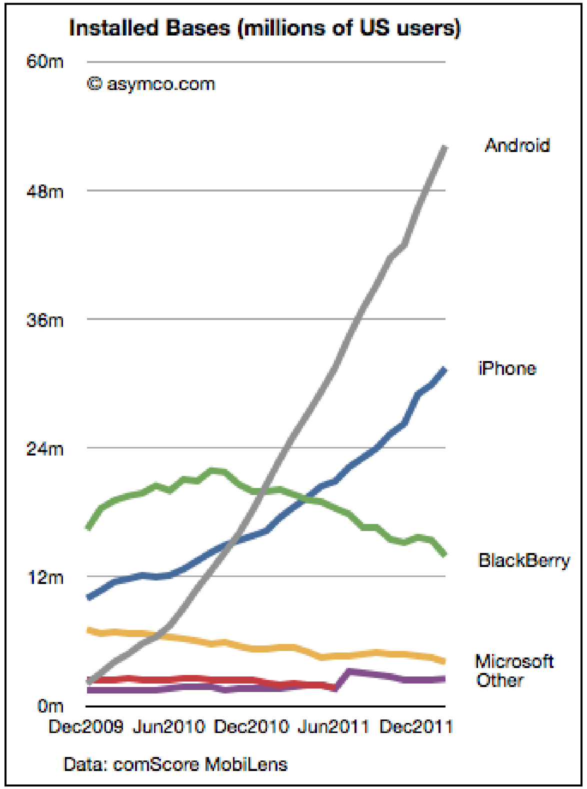
Example use of Khronos APIs in AR



OS API Adoption

OpenGL ES	**OpenGL ES 2.0** **Shipping - Android 2.2**
OpenSL ES	**OpenSL ES 1.0 (subset)** **Shipping – Android 2.3**
OpenMAX AL	**OpenMAX AL 1.0 (subset)** **Shipping - Android 4.0**
EGL	**EGL 1.4** **Shipping under SDK -> NDK**
WebGL	**Opera and Firefox WebGL now** **Chrome soon**
OpenGL	**OpenGL 3.2** **on MacOS**
OpenCL	**OpenCL 1.1** **on MacOS**
OpenGL ES	**OpenGL ES 2.0** **on iOS**
WebGL	**Can enable on MacOS Safari** **iOS5 enables WebGL for iAds**

Mobile Operating Systems

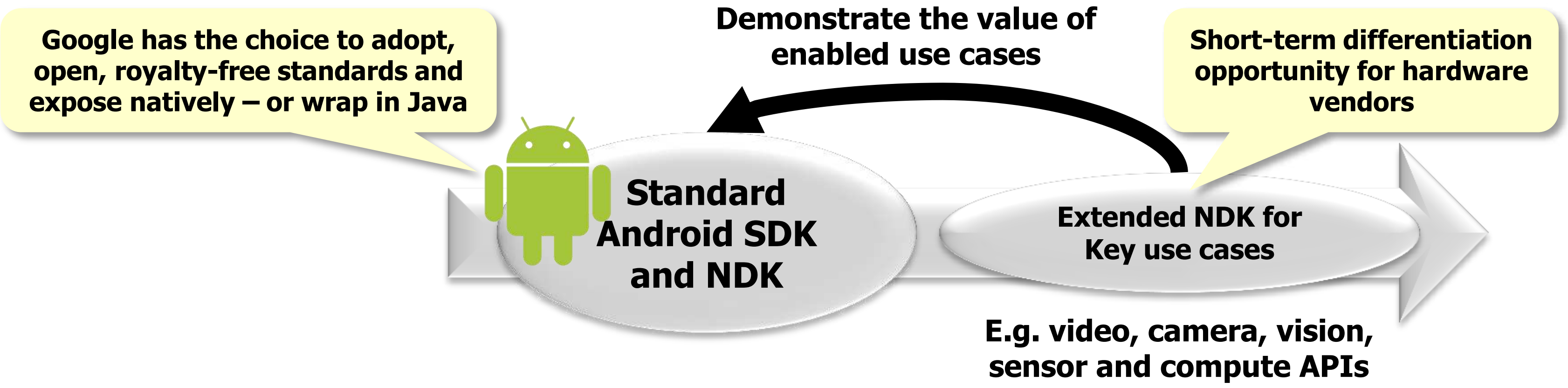


Microsoft Windows RT:

- **Only Microsoft native APIs**
- **HTML5 but not yet WebGL**

Extended Native APIs on Android

- **Native APIs can be shipped as NDK extensions before Google Adoption**
  - Do not break/change existing Google APIs
- **Khronos open standard APIs have strong momentum in silicon**
  - Google has option to adopt into standard platform to eliminate fragmentation
  - Exposed directly – or wrapped in Java binding
- **Extended APIs can be used by:**
  - Bundled apps, Market apps with API selection
  - Multiple APKs behind single multi-APK SKU



HTML5 – Cross OS App Platform

- **Increasing diversity of devices creates a demand for a true cross OS programming platform**
- **BUT need more than "more HTML"**

Traditional Web-content

Rich Experiential Processing

Multi-core CPUs
Rich 2D and 3D GPU
GPU Computing
Multiple HD cameras
Image and vision processing
Video encode/decode
Audio encode/decode
Inertial and positional sensors

How can the Browser rapidly assimilate such diverse functionality?

Leveraging Proven Native APIs into HTML5

- **Leverage native API investments into the Web**
  - Faster API development and deployment
  - Familiar foundation reduces developer learning curve
- **Khronos and W3C creating close liaison**
  - Multiple potential joint projects

HTML

Native APIs shipping or working group underway

JavaScript API shipping or working group underway

Possible future JavaScript APIs

WebGL – 3D on the Web – No Plug-in!

- **WebGL defines JavaScript binding to OpenGL ES 2.0**
  - Leveraging HTML 5 and uses <canvas> element
  - Enables a 3D context for the canvas
- **WebGL 1.0 Released at GDC March 2011**
  - Mozilla, Apple, Google and Opera working closely with GPU vendors
- **Low-level foundational API for accessing the GPU in HTML5**
  - Flexibility and direct GPU access - support higher-level frameworks and middleware

WebGL Implementation Anatomy

Content downloaded from the Web. Middleware can make WebGL accessible to non-expert 3D programmers

Content
JavaScript, HTML, CSS, ...

JavaScript Middleware

Browser provides WebGL functionality alongside other HTML5 specs - no plug-in required

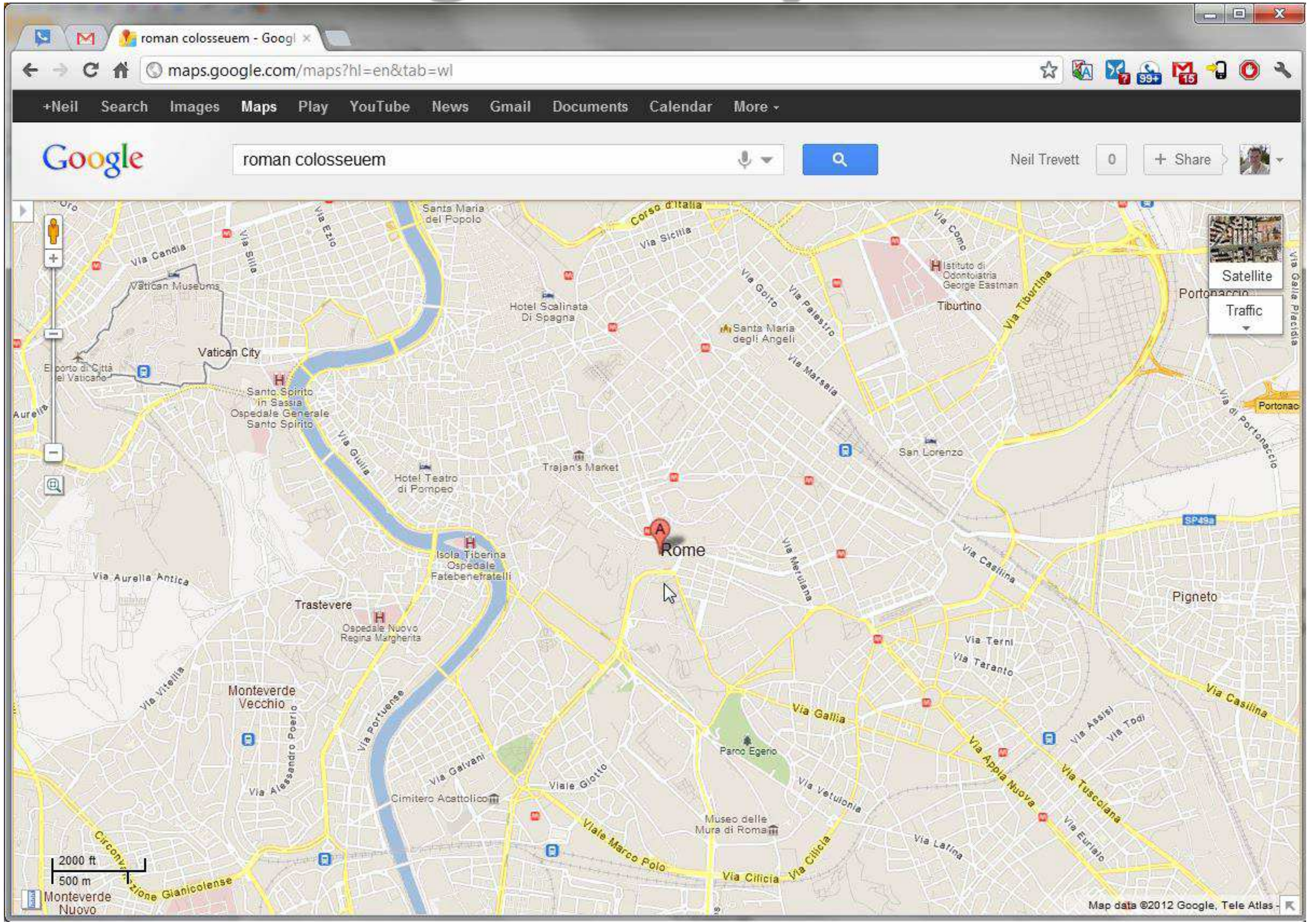
WebGL | **HTML5**

JavaScript | **CSS**

OS Provided Drivers. WebGL on Windows can use Google Angle to create conformant OpenGL ES 2.0 over DX9

OpenGL ES 2.0
OpenGL
DX9/Angle

WebGL – Being Used by Millions Every Day



WebGL and Security

- **WebGL is Architecturally Secure**
  - NO known WebGL security issues
  - Impossible to access out-of-bounds or uninitialized memory
  - Use of cross-origin images are blocked without permission through CORS
  - Browsers maintaining black lists - used if unavoidable GPU driver bugs discovered
- **DoS attacks and GPU hardening**
  - Draw commands can run for a long time -> unresponsive system
    - Even without loops in shaders
  - WebGL working closely with GPU vendors to categorically fix this
  - Short term: mandate ARB_robustness and associated GPU watchdog timer
  - **Longer term: GPUs need robust context switch and pre-emption**

WebCL – Parallel Computing for the Web

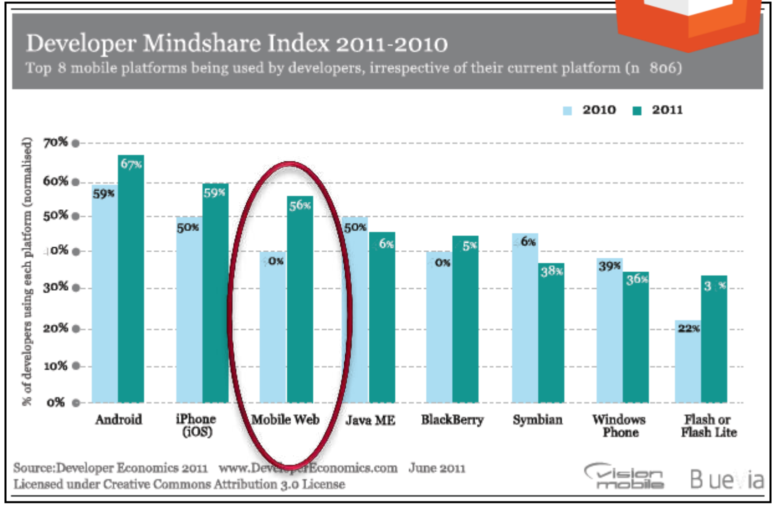
- **JavaScript bindings to OpenCL APIs**
  - JavaScript initiates OpenCL C Kernels on heterogeneous multicore CPU/GPU
- **Stays close to the OpenCL standard**
  - Maximum flexibility to provide a foundation for higher-level middleware
- **Minimal language modifications for 100% security and app portability**
  - E.g. Mapping of CL memory objects into host memory space is not supported
- **Compelling use cases**
  - Physics engines for WebGL games, image and video editing in browser
- **API definition underway – public draft just released**
  - https://cvs.khronos.org/svn/repos/registry/trunk/public/webcl/spec/latest/index.html

WebCL Demo

http://www.youtube.com/user/SamsungSISA#p/a/u/1/9Ttux1A-Nuc

Web Apps versus Native Apps

- **Mobile Apps have functional and aesthetic appeal**
  - Beautiful, responsive, focused
- **HTML5 with GPU acceleration can provide the same level of "App Appeal"**
  - Highly interactive, rich visual design
- **Using HTML5 to create 'Web Apps' has many advantages**
  - Web app is searchable and discoverable through the web
  - Portable to any browser enabled system
  - Same code can run as app or as web page
  - Not a closed app store – no app store 'tax'
- **How soon will we be able to write apps such as Augmented Reality in HTML5?**



Expanding Platform Reach for Graphics and Computation

Production Browsers Shipping with WebGL:
Desktop - Chrome, Firefox, Opera, Safari
Mobile - Opera and Firefox
Apple iOS Safari uses WebGL for iAds

	Desktop	Mobile	Web
Graphics	OpenGL	OpenGL ES	WebGL
	Interop	Interop, EGL	Interop, Typed Arrays
Compute	OpenCL Full Profile	OpenCL Full Profile *and* Embedded Profile	WebCL

WebGL on majority of production desktops now. WebGL pervasively available on mobile in next 12 months

WebCL will start deploying in next 12-18 months

OpenCL pervasively available on mobile in next 18-24 months

Cross-OS Portability

	Android	Apple	Microsoft Windows RT
HTML	HTML/CSS WebGL	HTML/CSS WebGL	HTML/CSS No WebGL
SDK	Dalvik (Java)	Objective C	C#
C/C++	OpenGL ES	OpenGL ES	DirectX

HTML5 provides cross platform portability. GPU accessibility through WebGL available soon on ~90% mobile systems

Preferred development environments not designed for portability

Native code is portable- but apps must cope with different available APIs and libraries

Summary

- Advances in SOC silicon processing and associated APIs are enabling significant new use cases
- Holistic cooperation between hardware and software needed to deliver increasing computational loads in a fixed power budget
- Architectural shifts, such as unified memory, are creating challenges and opportunities for applications and the APIs that enable them
- Mobile operating systems and HTML5 browsers both lag in exposing the latest SOC capabilities - creates functional differentiation opportunities
- Dynamic tension between platform vendors that want captive apps and developers that benefit from cross platform portability
- Cooperative API standards working hard to eliminate roadblocks to mobile industry growth

Thank You!

We will resume at 11:10AM