Topics

• Why open standards advance the industry
• How Khronos creates open API standards for silicon acceleration
• State of the art in visual computing APIs
• Vision and sensor processing for Augmented Reality
• Advanced hardware acceleration in HTML5
Why do we NEED Standards?

• **Need standard INTERFACES for INTEROPERABILITY** …
  … Widely adopted standard interfaces can grow market opportunity

• **Interoperability enables compelling user experiences** …
  … to be built cheaply enough to build a mass market

• **Standards avoid fragmentation that adds no value** …
  … Industries need to COOPERATE to build a market and then COMPETE

• **The mobile industry is define by standards**
  - GSM/EDGE, UMTS/HSPA, LTE, IEEE 802.11, Bluetooth, USB, HDMI …
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
Khronos Standards

15 Active Standards

- Embedded and Mobile 3D
- OpenGL
- OpenMAX
- OpenVG
- COLLADA
- OpenCL
- WebGL
- OpenVX
- Sensor Fusion
- Computer Vision
- Parallel Computing
- 3D Digital Asset Exchange format
- Advanced Audio
- Camera, Images and Streaming Media
- Vector 2D

Timeline:
- 2000
- 2003
- 2004
- 2005
- 2006
- 2008
- 2009
- 2011
Additional Khronos Initiatives

• **KITE**
  - Connecting Industry and Educators

• **Chapters**
  - Grass Roots Community for Khronos-related activities
  - Global network of like minded volunteers and participants – similar to ACM
  - Local networking, education and activities

![WebGL Meet-up San Francisco Sep11](image)
Khronos Working Groups

- Academic Members
  - Participation in WGs

- Contributor Members
  - Participation and vote in WGs

- Promoter Members
  - Participation and vote in WGs
  - Board seat for strategy, budget and spec ratification

API Working Groups (Industry and Academic)

- Conformance Tests and Adopters Program
- Ratified Specifications
- SDKs, Sample, Ref Cards and Man Pages
- Educator Guidelines and Courseware
- Certification Materials

KITE encourages academic participation in working groups to produce materials for educators and certification

Members
- Wider Community

Adopters
- Build conformant implementation and products

Developers
- Develop applications using the APIs

Educators / Certifiers
- Academic Courses
- Training and Certification

Contributor Members
- Participation and vote in WGs

Promoter Members
- Participation and vote in WGs
- Board seat for strategy, budget and spec ratification

Educator Guidelines and Courseware

Certification Materials

KITE Institute of Training & Education

© Copyright Khronos Group 2012 | Page 9
API Standards Evolution

New API technology first evolves on high-end platforms.

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

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

Diverse platforms – mobile, TV, embedded – mean HTML5 will become increasingly important as a universal app platform.
**OpenGL 20th Birthday - Then and Now**

Launched OpenGL 4.3 at SIGGRAPH 2012

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Triangles / sec (millions)</td>
<td>1</td>
<td>103 (x103)</td>
</tr>
<tr>
<td>Pixel Fragments / sec (millions)</td>
<td>240</td>
<td>1040 (x4.3)</td>
</tr>
<tr>
<td>GigaFLOPS</td>
<td>0.64</td>
<td>15.6 (x25)</td>
</tr>
</tbody>
</table>

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 ES

• Streamlined subset of desktop OpenGL for embedded and mobile devices

Fixed function 3D Pipeline

Programmable vertex and fragment shaders

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

GL4/DX11-class Capabilities come to ES ??

OpenGL 4.3 is a superset of DX11

GL4/DX11 - class
Capabilities come to ES ??

OpenGL ES 2.0
Platforms Released

OpenGL ES 3.0
Platforms Released

OpenGL ES 1.1
Platforms Released

OpenGL ES 1.1
Spec Released

OpenGL ES 1.0
Spec Released

OpenGL 1.5
OpenGL 2.0
OpenGL 2.1
OpenGL 3.1
OpenGL 3.2
OpenGL 3.3
OpenGL 4.0
OpenGL 4.1
OpenGL 4.2
OpenGL 4.3

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

- DXTC/S3TC
  - Windows
- PVRTC
  - iOS
- ETC1
  - Mandated in Android Froyo (400M devices)
- ETC2 / EAC
  - MANDATED in OpenGL ES 3.0
  - OpenGL 4.3
- ASTC
  - OpenGL ES 3.0 and OpenGL 4.3 Extension -> Core once proven

- Quality
- Deployment

- ETC1
  - Royalty-free
  - Backward compatible with ETC1
  - ETC2: 4bpp | 3 channel
  - EAC: 4 (8)bpp | 1(2) channel
  - COMBINE: RGBA 8bpp | 4 channel
  - Does not have 1-2 bit compression WITH ALPHA

- ETC2 / EAC
  - Royalty-free
  - BUT only optional in ES.
  - Only 4bpp | 3 channel
  - No alpha support

- ASTC
  - Royalty-free
  - Best quality.
  - Independent control of bit-rate and # channels
  - 1 to 4 channel
  - 1-8bpp in fine steps

- Royalty - free
- Platform Fragmentation

- 2008-2010
- 2012-2013
- 2014->
• **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

![Image of ladybug on yellow flowers with compression examples](image-url)
Kishonti GLBenchmark 3.0

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

Occlusion query used to determine light visibility

Instanced drawing used for vehicles and particles
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.
Processor Parallelism

- CPUs
  Multiple cores driving performance increases
  - Multi-processor programming – e.g. OpenMP

- GPUs
  Increasingly general purpose data-parallel computing
  - Graphics APIs and Shading Languages

Emerging Intersection

Heterogeneous Computing

OpenCL

OpenCL is a programming framework for heterogeneous compute resources
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 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

**Long-term Core Roadmap**
Significant enhancements to memory and execution model:
- Better handle irregular work loads
- Reduce overhead of host/device data exchange
- Better mage handing and API interop
- Enhanced language constricts and built-in functions for ease of use

**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 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 dot releases
  - Backwards compatibility protects software investment

- Diagram showing releases:
  - OpenCL 1.0 released Dec08
  - Conformance tests released Dec08
  - OpenCL 1.1 Specification and conformance tests released Jun10
  - OpenCL 1.2 Specification and conformance tests released Nov11
  - OpenCL 1.2 Specification update Nov12
  - OpenCL on mobile platforms begins to ship pervasively 2013/4
OpenCL 1.2 Update

• Create an OpenCL image from a OpenGL multi-sampled texture
  - Provides more flexibility in interoperating 3D graphics and compute

• Create 2D images from an OpenCL buffer
  - Process memory structures using the advanced properties of OpenCL images

• Security features for WebCL implementations layered over OpenCL
  - Initialize local and private memory before a kernel begins execution
  - Query and API to terminate an OpenCL context to ensure a long running kernel does not affect system stability

• Load an OpenCL program object from a Standard Portable Intermediate Representation (SPIR) instance
  - Increased tool chain flexibility and avoids the need to ship kernel source in commercial applications
OpenCL as Parallel Compute Foundation

- **HLM**: C++ syntax/compiler extensions
- **WebCL**: JavaScript binding to OpenCL for initiation of OpenCL C kernels
- **Aparapi**: Java language extensions for parallelism
- **River Trail**: Language extensions to JavaScript
- **C++ AMP**: C++ syntax/compiler extensions

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

Intel Shevlin Park Project using Clang/LLVM and OpenCL
http://llvm.org/devmtg/2012-11/#talk10
OpenCL and OpenGL Compute Shaders

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

1. Fine grain compute operations inside OpenGL
2. GLSL Shading Language
3. Execute on single GPU only

Developer driven decision

1. Full ANSI C programming of heterogeneous CPUs and GPUs
2. Utilize multiple processors
3. Coarse grain, buffer-level interop with OpenGL

Enhanced 3D Graphics apps “Shaders++”
Imaging Video Physics AI
Pure compute apps touching no pixels

Coming to mobile platforms soon!
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

**Sensor Fusion**
Devices become ‘magically’ context aware – location, usage, position

**Media and Image Streaming**

**Heterogeneous Parallel Processing**

**HTML5**
Web Apps that can be discovered on the Net and run on any platform
Visual-based Augmented Reality

Camera video stream sent to the compositor

GPS, accelerometer, gyro and camera images used to track the camera’s location and orientation

Camera Tracking

3D Augmentation Rendering

3D augmentations composited with video stream
OpenMAX - Media Acceleration

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

Portability and flexibility to access media components (codecs)

Sophisticated applications that provide their own media framework processing can interface to codecs directly

OpenMAX does not define CODECs - but provides codec interfaces

Portable access to native multimedia hardware acceleration
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
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

OpenMAX AL Media Player is the EGLStream “Producer” and controls production of frames.

EGLStreams enables and hides details of video frame transport. Enables multiple buffering modes for different uses cases e.g.: FIFO and explicit latch/release

OpenGL ES GL_TEXTURE_EXTERNAL is the EGLStream “Consumer” and converts video format into RGB OpenGL ES texture
How Many Sensors are in a Smartphone?

- Light
- Proximity
- 2 cameras
- 3 microphones (ultrasound)
- Touch
- Position
  - GPS
  - WiFi (fingerprint)
  - Cellular (tri-lateralation)
  - NFC, Bluetooth (beacons)
- Accelerometer
- Magnetometer
- Gyroscope
- Pressure
- Temperature
- Humidity

19!
Market Demand for Sensor Fusion API

- Innovative use of growing sensor diversity
- Effective use of multiple interoperating sensors in one app
- PORTABLE apps need to be isolated from sensor details
- Application developers do not wish to be Sensor Fusion experts
- Do NOT force the application developer to access individual sensors (unlike almost all other sensor APIs)
- High-level API enables sensor vendors to drive and deliver competitive sensor fusion innovation
Portable Access to Sensor Fusion

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

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

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

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

Spec expected in 2013
OpenVX

- **Vision Hardware Acceleration Layer**
  - Enables hardware vendors to implement accelerated imaging and vision algorithms
  - For use by high-level libraries or apps

- **Focus on enabling real-time vision**
  - On mobile and embedded systems

- **Diversity of efficient implementations**
  - From programmable processors, through GPUs to dedicated hardware pipelines

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

- Aiming for provisional specification in 1H 2013
  - Final specification 2H13
- Itseez is working group chair
OpenVX Execution Flow

- **OpenVX Graph for efficient execution**
  - Each Node can be implemented in software or accelerated hardware

- **EGL provides data and event interop – with streaming**
  - BUT use of other Khronos APIs are not mandated

- **VXU Utility Library provides efficient access to single nodes**
  - Open source implementation
Example use of Khronos APIs in AR

- Positional Sensors
  - Positional and GPS Sensor Data
  - Synchronization and sensor fusion
    - Computer Vision/Tracking & Computational Photography
      - Tracked features
      - Position and Tracking Semantics
- Camera
  - Camera Control
    - Preprocess and generate video streams
  - EGLStream
    - Image stream to GPU
  - 3D Rendering and Video Composition On GPU
- OpenSL|ES
- OpenVX
- OpenMAX|AL
- EGL
- OpenCL
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

How can the Browser rapidly assimilate such diverse functionality?

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
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 exploring liaison
  - Multiple potential joint projects
WebGL – 3D on the Web – No Plug-in!

• Historic opportunity to bring accelerated 3D graphics to web
  - 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
WebGL Implementation Anatomy

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

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

OS Provided Drivers. WebGL on Windows can use Google Angle to create conformant OpenGL ES 2.0 over DX9.
WebGL – Being Used by Millions Every Day
WebGL and Security

• **WebGL is Architecturally Secure**
  - Currently, 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: GPU provides increasingly robust security and multi-tasking
WebCL – Parallel Computing for the Web

• **JavaScript bindings to OpenCL APIs**
  - Enables initiation of Kernels written in OpenCL C within the browser

• **Bindings stay 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

• **API definition underway – public draft released**

• **Compelling use cases**
  - Physics engines for WebGL games
  - Image and video editing in browser
WebCL Demo

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

WebCL for Hardware-Accelerated Web Applications

Advanced Browser Technology
Samsung R&D Center
San Jose, CA
Expanding Platform Reach for Graphics and Computation

**Desktop**
- **Graphics**: OpenGL
- **Compute**: OpenCL
  - Full Profile

**Mobile**
- **Graphics**: OpenGL ES
- **Compute**: OpenCL
  - Full Profile and Embedded Profile

**Web**
- **Graphics**: WebGL
- **Compute**: WebCL
  - Interop
  - Typed Arrays

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

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
API Adoption

**OpenGL ES 2.0**
Shipping - Android 2.2

**OpenSL ES 1.0**
Shipping – Android 2.3

**OpenMAX AL 1.0**
Shipping - Android 4.0

**EGL 1.4**
Shipping under SDK -> NDK

**WebGL**
Chrome on ICS will have WebGL. Opera and Firefox WebGL now

**OpenGL 3.2**
on MacOS

**OpenCL 1.1**
on MacOS

**OpenGL ES 2.0**
on iOS

Can enable on MacOS Safari

**iOS5** enables WebGL for iAds

---

**Mobile Operating Systems**

<table>
<thead>
<tr>
<th>Installed Bases (millions of US users)</th>
</tr>
</thead>
<tbody>
<tr>
<td>60m @ asymco.com</td>
</tr>
<tr>
<td>48m</td>
</tr>
<tr>
<td>36m</td>
</tr>
<tr>
<td>24m</td>
</tr>
<tr>
<td>12m</td>
</tr>
<tr>
<td>0m</td>
</tr>
</tbody>
</table>

Data: comScore MobileLens

**Microsoft WinRT/Metro:**
- 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 APIs have strong momentum in the silicon community
  - Evangelize Google to adopt into standard platform
- Extended APIs can be used by:
  - Bundled apps, Market apps with API selection
  - Multiple APKs behind single multi-APK SKU

Evangelize Google and ISVs to adopt enabling APIs

E.g. video, camera, imaging, sensor and composition APIs

Standard Android SDK and NDK

Extended NDK for Key use cases
In Summary

- APIs are key to enable compelling applications on advanced hardware – APIs developed on high-end hardware are now enabling mobile devices
- APIs no longer exist alone – they interoperate and provide input AND output processing to form a complete platform for advanced content
- Significant cooperation underway between native and Web APIs to bring advanced visual computing to HTML5
- Khronos is driving open standards for hardware acceleration
  Participate, change the industry AND get the inside edge for your products!