Overview of the OpenVG, a vector graphics API
and
Introduction to IWAYAG, an native OpenVG accelerator

December 18, 2009

Hiroshi Fujiwara
NEC System Technologies, Ltd.
Agenda

Overview of the OpenVG, a vector graphics API
- Graphics models. Summary and Comparison
- OpenVG Key features, Eco-System and Market
- OpenVG Pipeline Design
- OpenVG Implementations
- OpenVG1.1

Introduction to IWAYAG, a native OpenVG accelerator
- IWAYAG Features
- Implementation Examples of IWAYAG
- IWAYAG Performance
- IWAYAG Eco-System
- Loadmap
- Demonstration at Khronos booth
Overview of the OpenVG, a vector graphics API
Summary of typical graphics models

**Presentation Quality**

<table>
<thead>
<tr>
<th>3D Graphics</th>
<th>Text Drawing</th>
<th>Shape Drawing</th>
<th>Image Drawing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vector font (Outline font)</td>
<td>Convert Bezier curve to polygon</td>
<td>Polygons with Anti-Alias</td>
<td>Textures</td>
</tr>
<tr>
<td>Realistic Expression</td>
<td>Realistic Expression</td>
<td>Realistic Expression</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Vector Graphics</th>
<th>Text Drawing</th>
<th>Shape Drawing</th>
<th>Image Drawing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vector font (Outline font)</td>
<td>Direct Bezier Curve drawing</td>
<td>Bezier Curves with Anti-Alias</td>
<td>Perspective Transformation</td>
</tr>
<tr>
<td>Animation-type Expression</td>
<td>Semi-3D Expression</td>
<td>Realistic Expression</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2D Graphics</th>
<th>Text Drawing</th>
<th>Shape Drawing</th>
<th>Image Drawing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bitmap font</td>
<td>Lines, Circles</td>
<td></td>
<td>Bitmaps</td>
</tr>
</tbody>
</table>
Cost-Performance comparison between models

Cost to realize rich graphics by each approach

<table>
<thead>
<tr>
<th>Approach</th>
<th>Graphics HW Cost</th>
<th>Required CPU</th>
<th>Suitability</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D</td>
<td>☎ Small footprint Low Power</td>
<td>✗ High Performance Expensive, High Power</td>
<td>✗ Poor</td>
</tr>
<tr>
<td>Vector</td>
<td>○ Medium footprint Low Power</td>
<td>○ Low Performance, Low cost, Low power</td>
<td>☎ Good overall balance</td>
</tr>
<tr>
<td>3D</td>
<td>✗ Large footprint High Power</td>
<td>○ Low Performance, Low cost, Low power</td>
<td>○ Over spec?</td>
</tr>
</tbody>
</table>
OpenVG – Accelerated Vector Graphics

- **OpenVG is the industry’s first native Bezier rendering API**
  - Enables a new class of hardware acceleration – Bezier primitives – not polygons
    - Smaller command/data list and no tessellation required, allowing such devices that have limited CPU/memory resources to enjoy rich graphics

- **Primarily used to accelerate higher-level engines such as Flash and SVG**
  - OpenVG is an OpenGL-style, low-level API – but many artists prefer tools over programming

- **Support set of primitives to realize high quality graphics**
  - Gradation (A)
  - Alpha Blending (B)
  - Projective Transformation (C)
  - Vector font (D)
  - Anti-Alias (E)
OpenVG Eco Systems

- FlashLite 3.1.5 is the most popular Open-VG app (middle ware) as well as vector fonts
- 2D graphics library like Cairo can be ported on OpenVG, allowing WebKit to be accelerated
- XAML player can also take advantage of OpenVG
RichUI is becoming a MUST in Embedded Devices. OpenVG is wide-spread to these devices.

- Apple introduced iPhone featuring touch screen based rich user interface to mobile phone
- User interface was recognized as one of key success factors
- Many mobile phone vendors followed Apple, introducing the touch screen based UI
- Rich UI market is extending to car navigation devices, digital cameras, TVs, etc

OpenVG is getting widely accepted as a graphics API for the embedded devices.
OpenVG pipeline design

Like other Khronos APIs, OpenVG has been designed to enable software to effectively use silicon.

Pipeline has also been architected as “silicon-ready” design.
OpenVG Implementations

- There are a wide range of OpenVG implementations possible
  - Depending on the price point and target device architecture

- 3D GPUs can accelerate both OpenVG and OpenGL ES APIs
  - Good approach for those devices requiring true 3D and VG

- Dedicated VG accelerator allows higher performance rendering in lower cost than 3D GPUs
  - Good approach for those devices that do NOT require true 3D
OpenVG 1.1

- OpenVG 1.1 specification publicly released at SIGGRAPH Asia 2008
  - Together with open source sample implementation and full Conformance Test Suite

- OpenVG 1.1 Conformance Test
  - Comprehensive set of tests (~900 test cases)
  - Conformant implementation has to pass all of them

- Major features
  - Adobe Flash 7 / Flash Lite 3 support
  - Glyph API for hardware accelerated text rendering
  - Multi-sampled anti-aliasing
Introduction to IWAYAG, a native OpenVG accelerator
History of Graphics products at NEC System Technologies

2D Graphics Products

- Standard Graphics
  - ECA3
  - EMA
  - SSP
  - ELI
  - VEC
- Lowend 2D Graphics
  - ECA3
  - EMA
  - SSP
  - ELI
  - VEC
- Video accelerator
  - TPL
- LCD support
  - LIGHT
- Notebook

3D Graphics Products

- Fast2D Graphics
  - EXB
  - TADARE
  - ARET
  - CCE
  - ZBC
  - GPC
  - JAB
  - FIRE
  - ROE3A
  - WBA
  - COA3
  - CDC
  - NGPC
  - TE1
  - TE2
  - TE3
  - TE3V
  - TE3A
  - TE4
  - TE4Pro
  - TE5

GA88 Series

Graphics IP Products

Legend:

- : LSI
- : Board products
What is IWAYAG?

IWAYAG is a graphics accelerator dedicated for vector graphics, consisting of:
- Hardware IP core to be incorporated to the customer’s SoC
- OpenVG driver

IWAYAG design and features

- Native OpenVG accelerator
  - Architecture of IWAYAG is based on the OpenVG pipeline stage design
  - Path generation to Blending/Anti-alias stages are implemented by hardware
  - No CPU tessellation. Fill is done by hardware using outline buffer
- Hardware bezeir rasterizer
- Extensive cache control mechanism for fast filling
- Support fast FSAA

Path generation to Blending/Anti-alias stages are implemented by hardware.
Why IWAYAG?

Requirements from the consumer devices market

- Flash support
  - Many consumer device manufacturers plan to incorporate richer UI
  - Flash support is a MUST to efficiently develop the rich UI
- Smooth playback of Flash contents while using low-end CPUs
  - Cost and/or battery power limits CPU selection
    - 100Mhz CPU without FPU is common for Digital cameras.
- Long battery life
  - Flash acceleration by 3D engine consumes too much battery powers

Native OpenVG graphics accelerators best meet the requirements

- Good Flash performance while using low-end CPUs
- Much lower power and die size than 3D engine
Implementation Example of IWAYAG
(1) Xilinx ML510 (PowerPC440 based FPGA board)

-CPU is PowerPC440 @ 300MHz
-Though IWAYAG has been designed to run at 200MHz clock, this implementation drives IWAYAG at 50MHz due to the limitation of the board

Performance data demonstrated in the following pages were measured by executing Flash contents both in software renderer (disabling IWAYAG) and hardware renderer (enabling IWAYAG) on the same ML510 board.
Performance Improvements on Flash Contents by IWAYAG (1)

**(1)002_motionTweenLine.swf**

<table>
<thead>
<tr>
<th>Software</th>
<th>IWAYAG 50MHz</th>
<th>Acceleration ratio = 3.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>IWAYAG 200MHz (Estimated)</td>
<td></td>
<td><strong>Acceleration Ratio = 3.0-7.3</strong></td>
</tr>
</tbody>
</table>

**Average execution time per frame (ms)**

**Fast**

**(2)003_motionTweenGradient.swf**

<table>
<thead>
<tr>
<th>Software</th>
<th>IWAYAG 50MHz</th>
<th>Acceleration ratio = 3.8</th>
</tr>
</thead>
<tbody>
<tr>
<td>IWAYAG 200MHz (Estimated)</td>
<td></td>
<td><strong>Acceleration Ratio = 3.8-11.5</strong></td>
</tr>
</tbody>
</table>

**Average execution time per frame (ms)**

**Fast**

**(3)004_motionTweenBMP.swf**

<table>
<thead>
<tr>
<th>Software</th>
<th>IWAYAG 50MHz</th>
<th>Acceleration ratio = 3.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>IWAYAG 200MHz (Estimated)</td>
<td></td>
<td><strong>Acceleration Ratio = 3.5-8.9</strong></td>
</tr>
</tbody>
</table>

**Average execution time per frame (ms)**

**Fast**
Performance Improvements on Flash Contents by IWAYAG (1)

<table>
<thead>
<tr>
<th>File Name</th>
<th>Description</th>
<th>Frame Rate</th>
<th>Acceleration Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>(4)005_motionTweenText.swf</td>
<td>Software</td>
<td>2.9</td>
<td>2.9-7.8</td>
</tr>
<tr>
<td></td>
<td>IWAYAG 50MHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>IWAYAG 200MHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(5)006_motionTweenMC.swf</td>
<td>Software</td>
<td>2.4</td>
<td>2.4-6.7</td>
</tr>
<tr>
<td></td>
<td>IWAYAG 50MHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>IWAYAG 200MHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(6)007_shapeTween.swf</td>
<td>Software</td>
<td>3.4</td>
<td>3.4-9.1</td>
</tr>
<tr>
<td></td>
<td>IWAYAG 50MHz</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>IWAYAG 200MHz</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Performance Improvements and CPU Usage analysis

Software Rendering

<table>
<thead>
<tr>
<th>Application</th>
<th>Graphics</th>
<th>- CMD/DATA Setup</th>
</tr>
</thead>
<tbody>
<tr>
<td>Video</td>
<td>- Path generation</td>
<td></td>
</tr>
<tr>
<td>Network</td>
<td>- Coordinates</td>
<td></td>
</tr>
<tr>
<td>I/Os</td>
<td>- Transformation</td>
<td></td>
</tr>
<tr>
<td>Application</td>
<td>- Rasterize</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Paint</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- Image</td>
<td></td>
</tr>
<tr>
<td>CPU</td>
<td>- Blending</td>
<td></td>
</tr>
</tbody>
</table>

IWAYAG

Vector Graphics Engine

Rasterizer

(w/cache control)

Clipping

Masking

Bus controller

Alpha blending

Graphics

processing

Offloaded to

IWAYAG

Performance

CPU Usage

| CPU+IWAYAG | Performance: 43fps | CPU Usage: 39% |

3.3 times better
Performance by
55% CPU Usage

X 3.3

△55%

Measured by Adobe FlashLite
With Motion_Tween_BMP.swf
Performance versus CPU clock

Software

004_motionTweenBMP.swf

Software renderer performs in direct proportion to CPU clock

HWAYAG 50MHz

Hardware renderer is almost independent to CPU clock

Enables high FPS rate with Low power CPU devices
Implementation Example of IWAYAG
(2) NEC EP-3 SiP (System in a Package)

IWAYAG in EP-3 is a solution for the customers to develop OpenVG-ready custom micro controllers in much lower development cost and time than SOC.

The EP-3 from NEC Electronics is a latest SiP (System in a Package) device, which features a 32-bit CPU and a user logic (gate array) in single package.

With EP-3, the customer can create custom microcomputers mounted with a CPU core and custom logic, by designing and developing only a Gate Array. Therefore the development cost and time can be drastically reduced compared with SoC design.

IWAYAG in EP-3 allows the customers to develop OpenVG-ready custom micro controllers in low development cost and time. IWAYAG in EP-3 is ideal solution for those brown goods, industrial devices and OA devices that the production volume is relatively small and life cycle is relatively short.

<table>
<thead>
<tr>
<th>Technology</th>
<th>Device Cost</th>
<th>Development Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPGA</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>SiP</td>
<td>Fair</td>
<td>Fair</td>
</tr>
<tr>
<td>SoC</td>
<td>Low</td>
<td>High</td>
</tr>
</tbody>
</table>
Implementation Example of IWAYAG (2) NEC EP-3 SiP (System in a Package)

In this prototype implementation, EP-3 evaluation board which consists of CPU and FPGA is used. IWAYAG is installed in the FPGA.
The user interface design is becoming a key success factor in many devices. We are supporting these demands extending the eco-system of IWAYAG to meet various needs from the customers.

- FlashLite and Outline font are already available
- Planning to support XAML player, 2D graphics library like Cairo and HTML renderer like Webkit
- Gesture UI engine is under planning as a companion IP for the next generation UI
Loadmap

2007

OpenVG1.0

2008

OpenVG1.1

2009

Lowend products
- Smaller gate size
- Lower power consumption
- Lower IP cost

Highend products
- Improved OpenVG acceleration
- Support future versions of OpenVG
- 3D support (OpenVG+3D hybrid)
- Support variety of display devices
- Easier customization

Coming Soon!

Extends the rich UI by OpenVG to wider range of the devices
Demonstration at Khronos booth

Booth: C12-⑤
Empowered by Innovation

All product names are trademarks or registered trademarks of their respective holders.