SEGA

Dreamcast Developer's Conference

Day 2
March 20, 1999

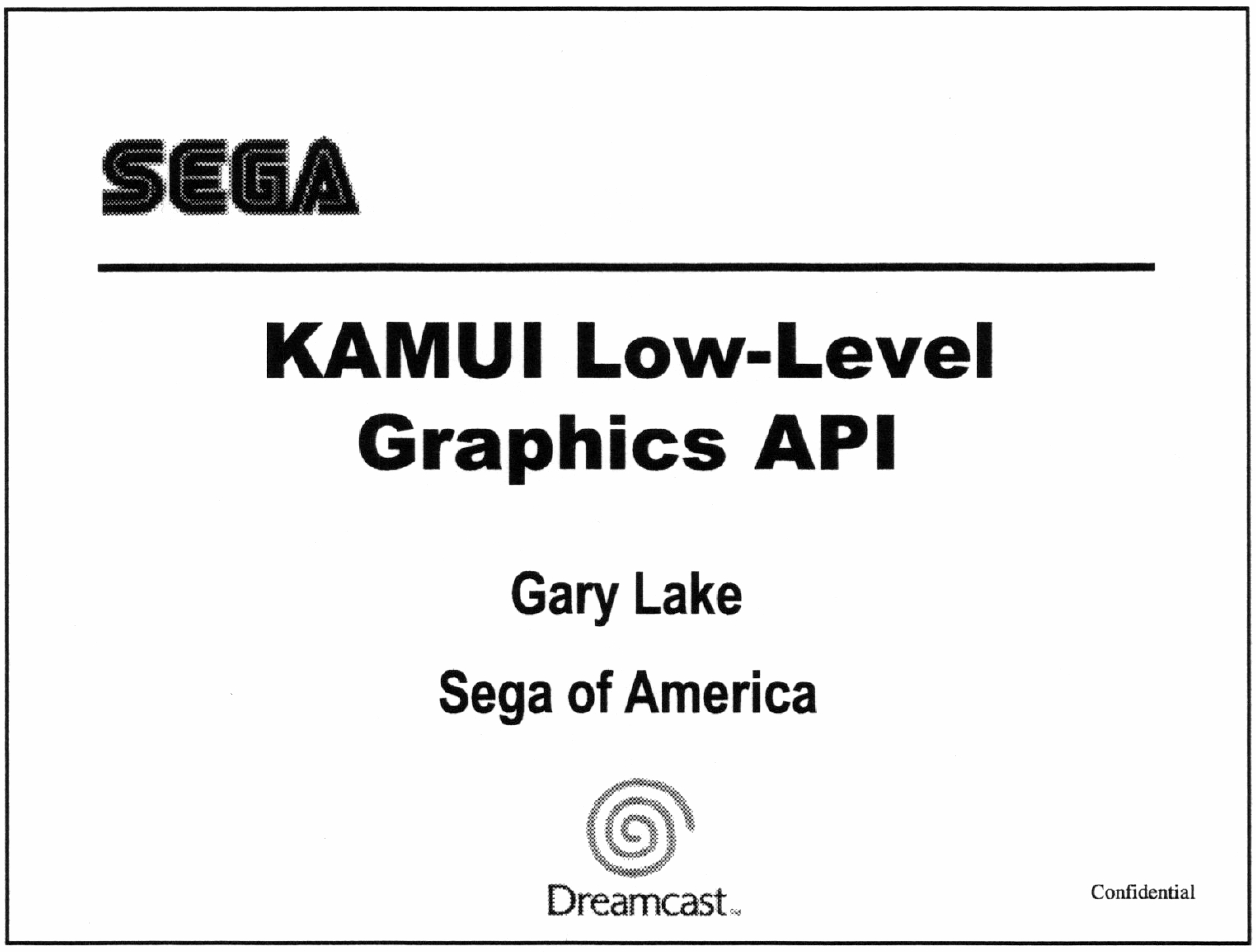
Dreamcast™

SEGA

Kamui Low-Level Graphics API

Gary Lake
Sega of America

Dreamcast™



SEGA

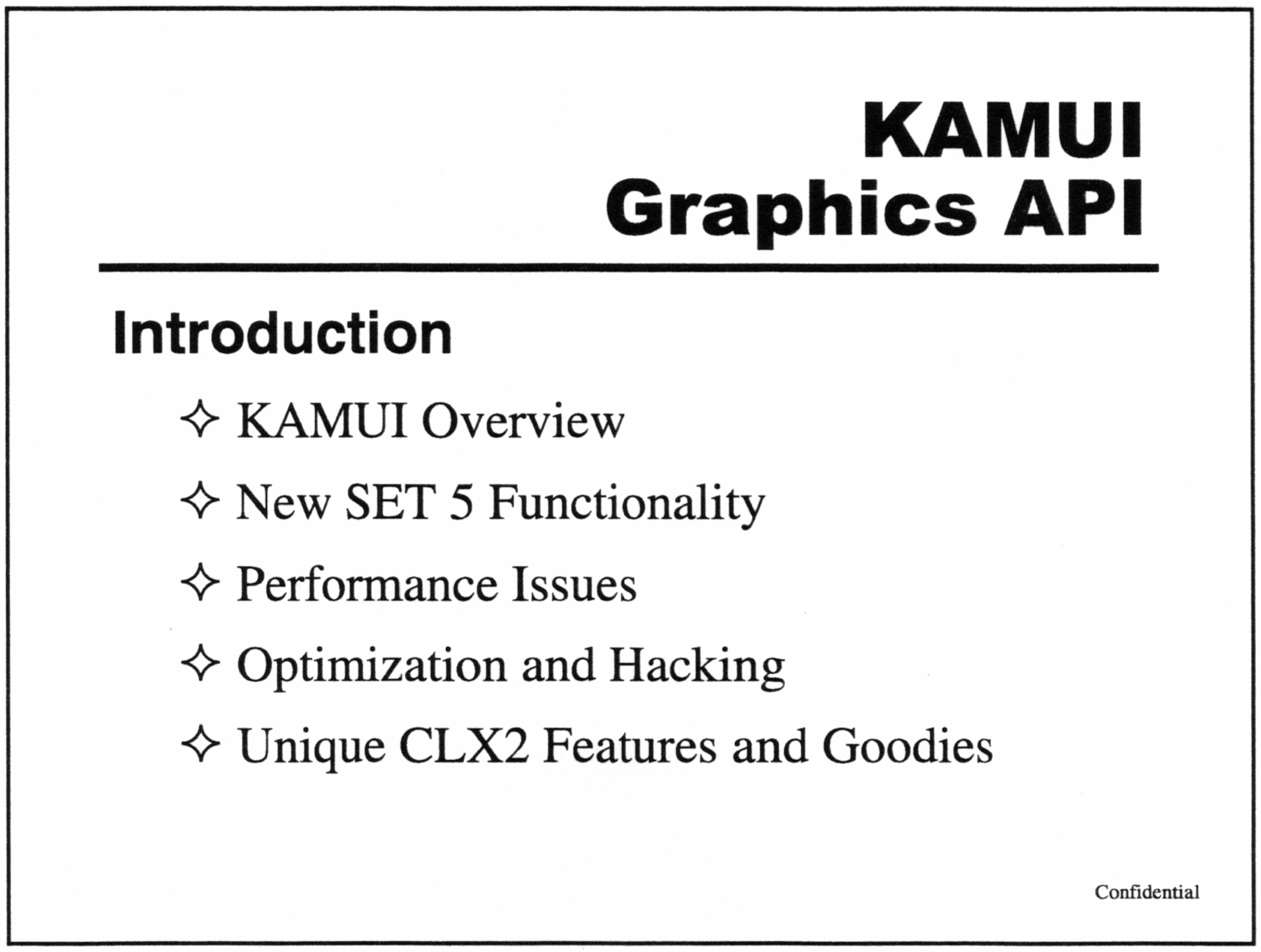
KAMUI Low-Level Graphics API

Gary Lake

Sega of America

Dreamcast

Confidential

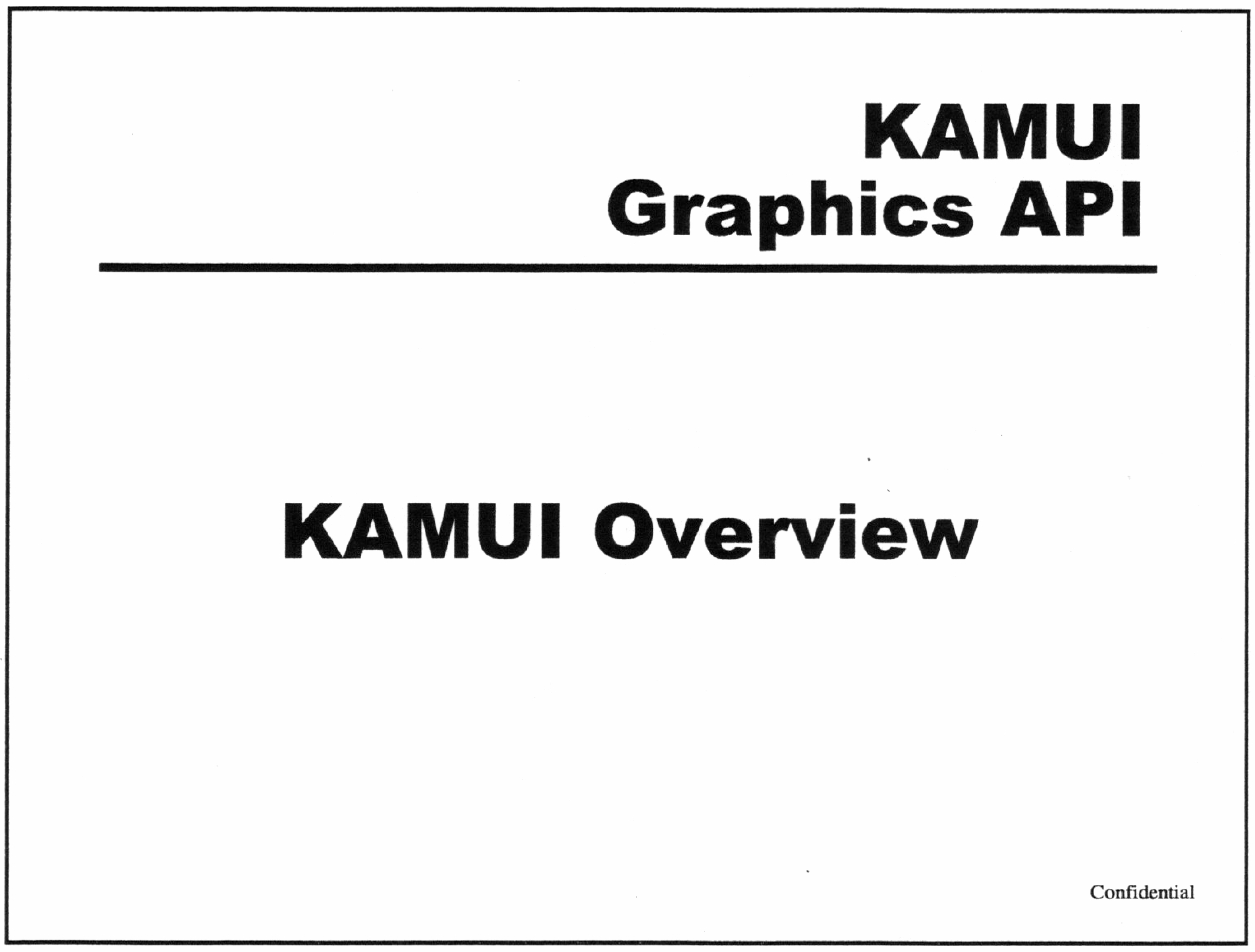


KAMUI Graphics API

Introduction

- ✧ KAMUI Overview
- ✧ New SET 5 Functionality
- ✧ Performance Issues
- ✧ Optimization and Hacking
- ✧ Unique CLX2 Features and Goodies

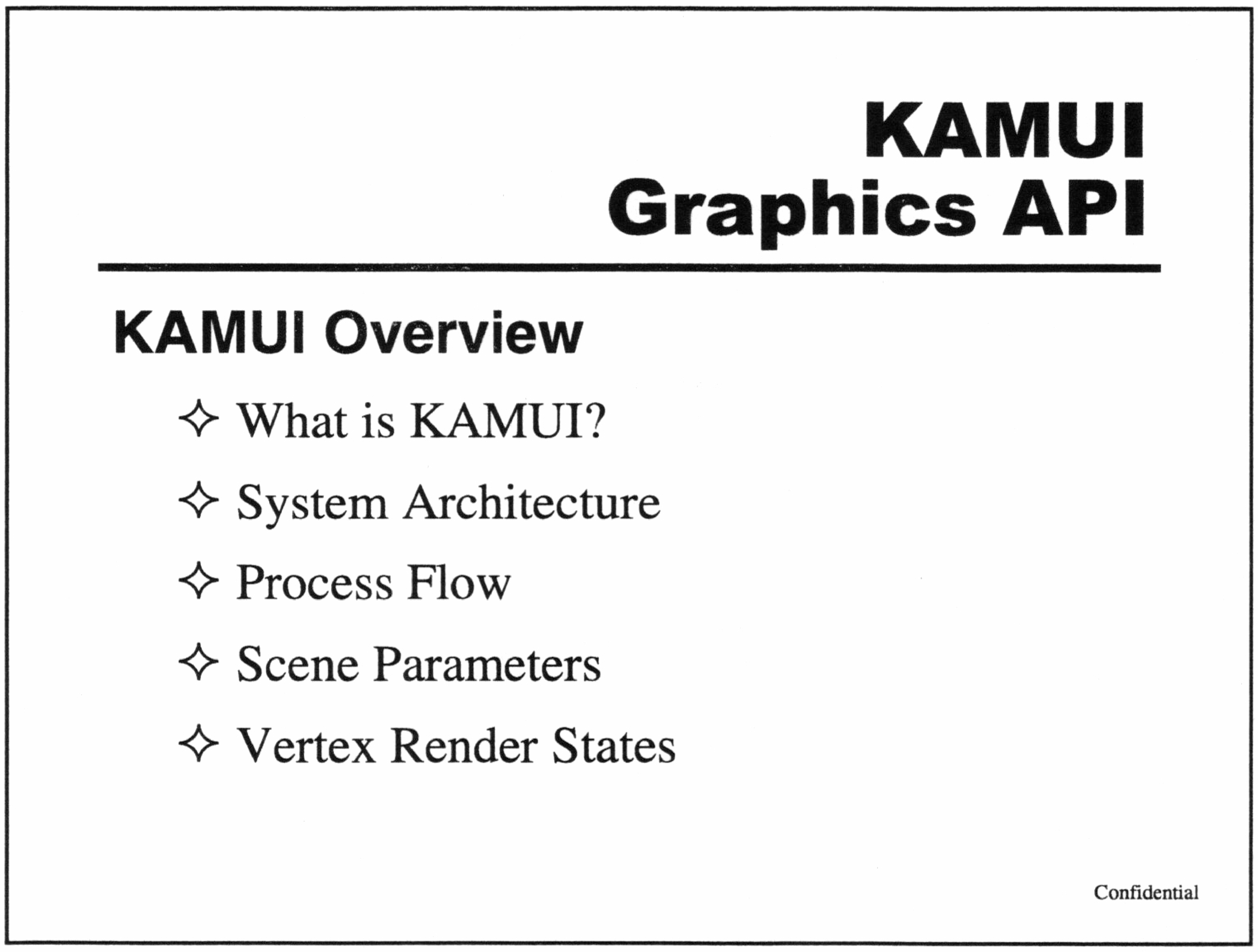
Confidential



KAMUI Graphics API

KAMUI Overview

Confidential

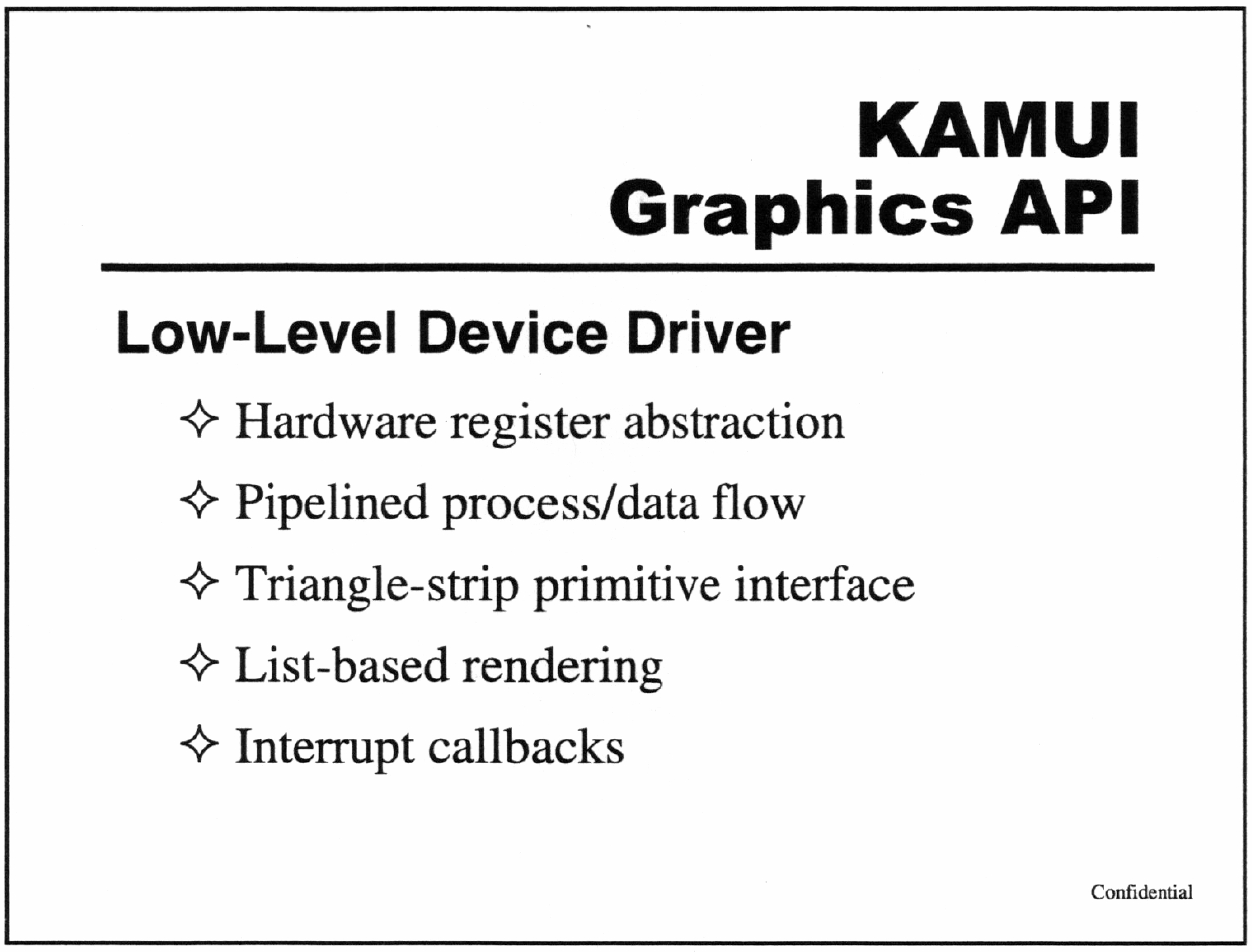


KAMUI Graphics API

KAMUI Overview

- ✧ What is KAMUI?
- ✧ System Architecture
- ✧ Process Flow
- ✧ Scene Parameters
- ✧ Vertex Render States

Confidential

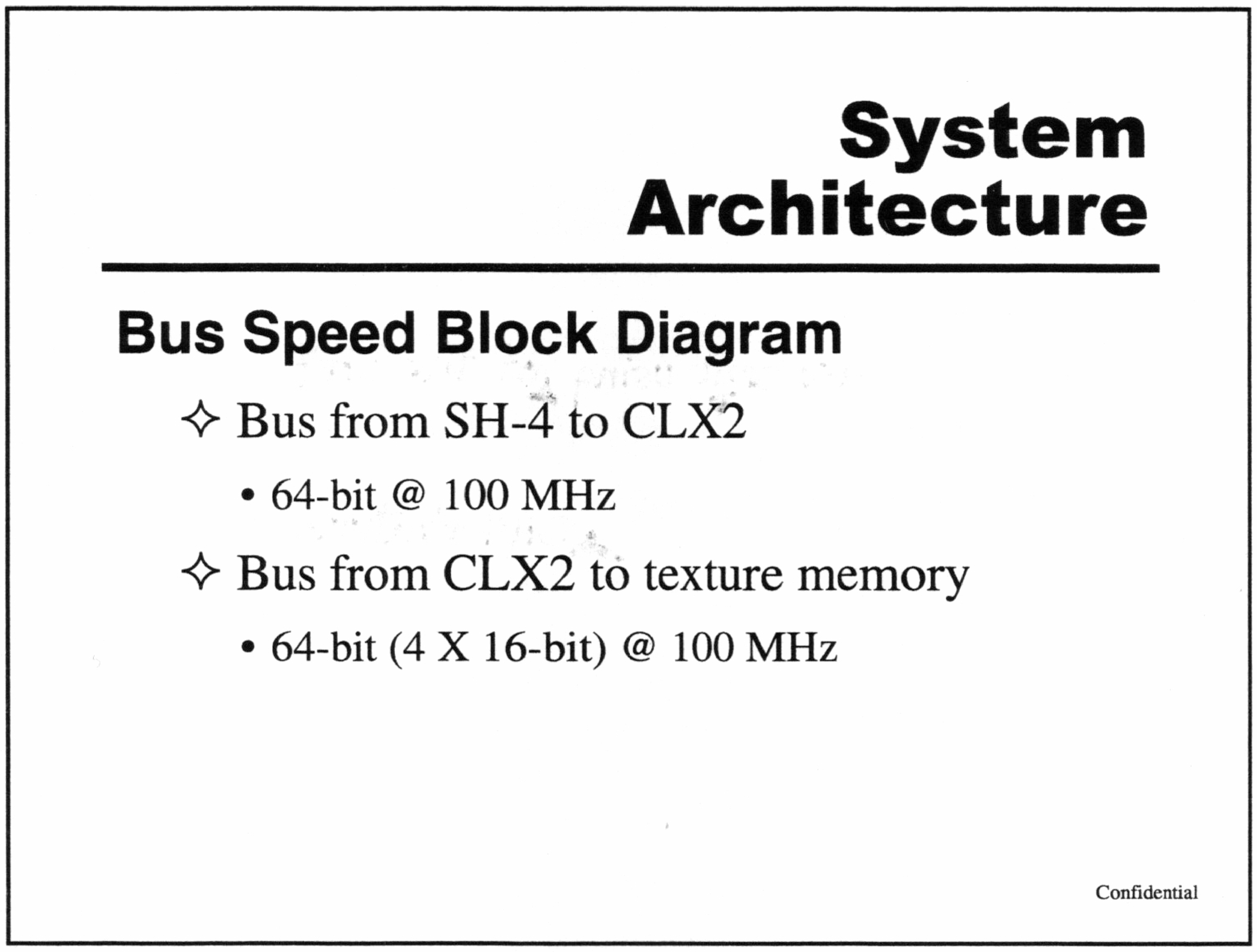


KAMUI Graphics API

Low-Level Device Driver

- ✧ Hardware register abstraction
- ✧ Pipelined process/data flow
- ✧ Triangle-strip primitive interface
- ✧ List-based rendering
- ✧ Interrupt callbacks

Confidential



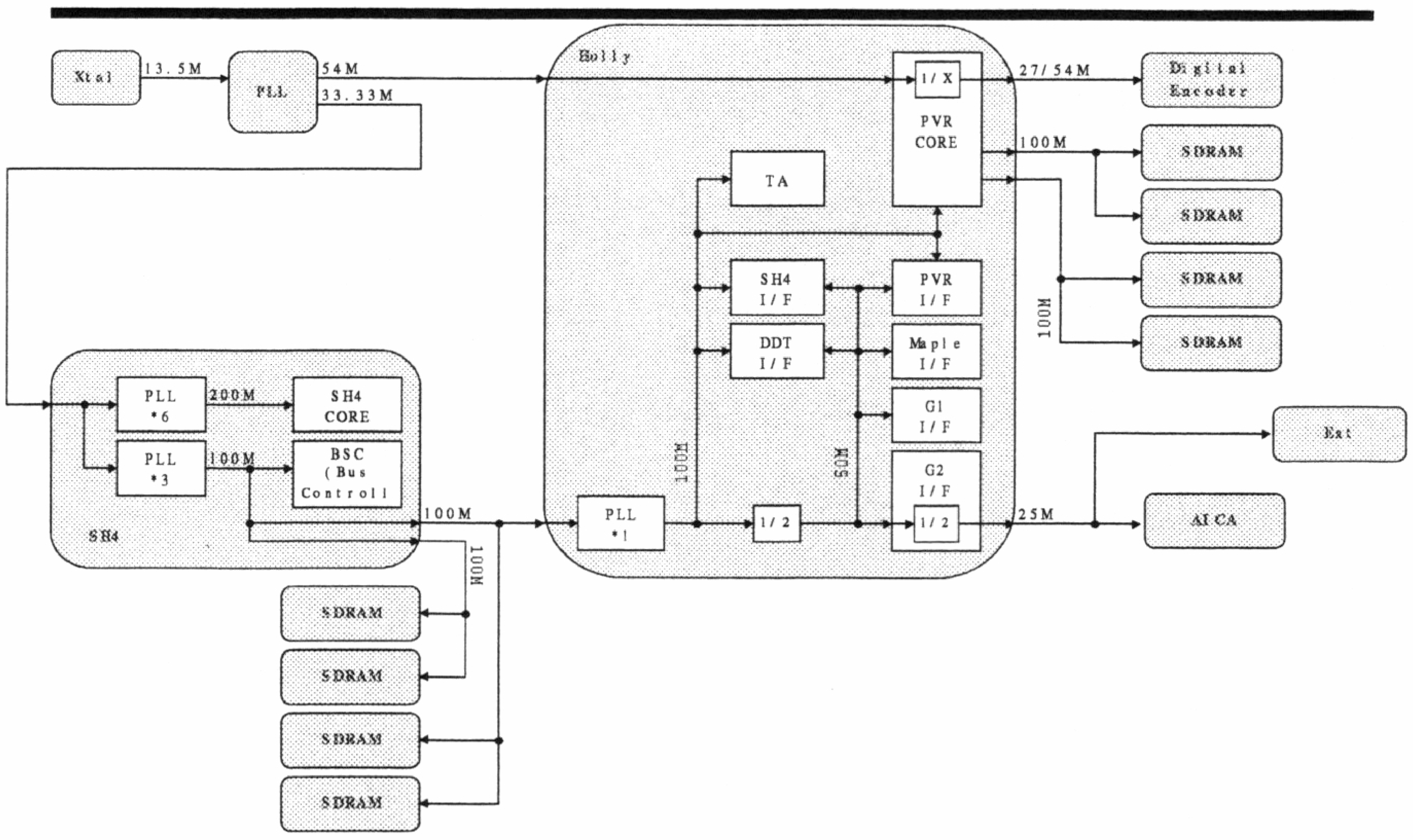
System Architecture

Bus Speed Block Diagram

- ✧ Bus from SH-4 to CLX2
  - 64-bit @ 100 MHz
- ✧ Bus from CLX2 to texture memory
  - 64-bit (4 X 16-bit) @ 100 MHz

Confidential

Bus Speed Block Diagram



System Architecture

Graphics Data Flow

are sent using of the SH4

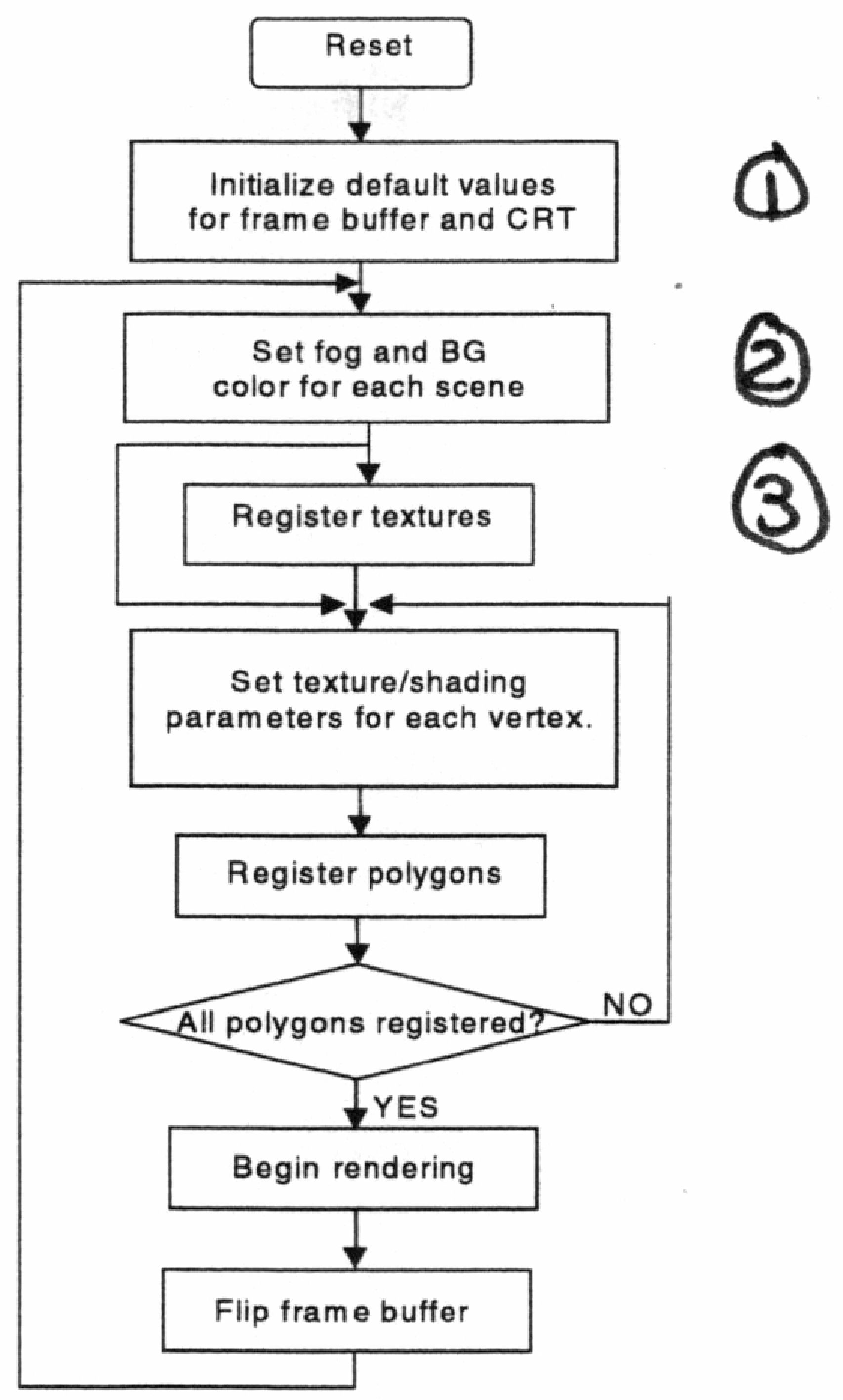
- ✧ Texture loads / DMA (800 MB/s peak)
- ✧ Vertex registration

  SH4 feature

  - DMA or Store Queue vertex data and control parameters to Tile Accelerator
- ✧ Scene render / CLX2 texture bus

Confidential

KAMUI Process Flow Diagram


Reset
Initialize default values for frame buffer and CRT
①
Set fog and BG color for each scene
②
Register textures
③
Set texture/shading parameters for each vertex.
Register polygons
All polygons registered?
NO
YES
Begin rendering
Flip frame buffer

KAMUI Process Flow

KAMUI Process Flow Diagram

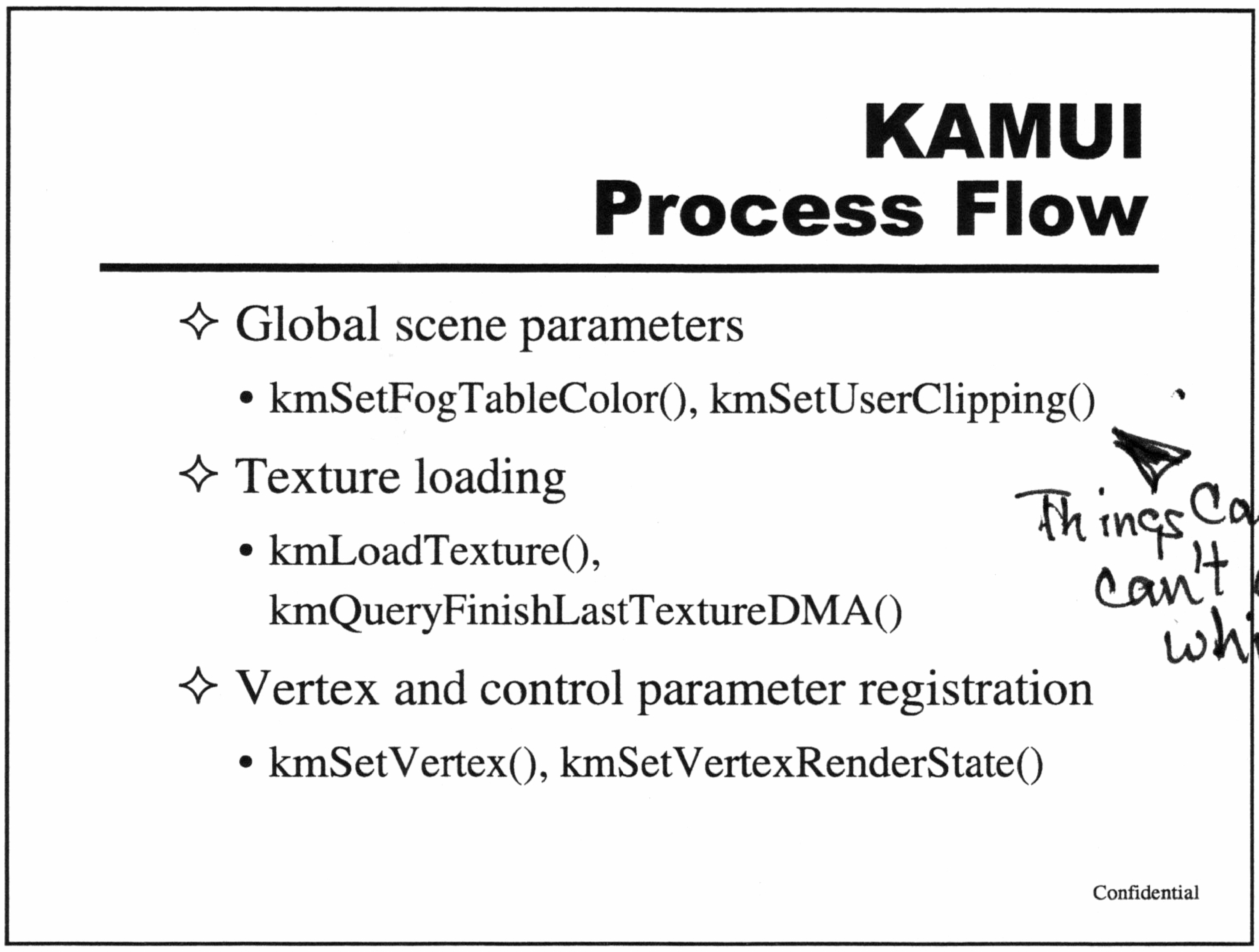
- ✧ System and graphics initialization
  - sbInitSystem(), kmInitDevice(), kmSetDisplayMode()

Init Shinobi

- ✧ Rendering system initialization
  - kmGetVersionInfo(), kmSetSystemConfig(), kmProcessVertexRenderState()

Where vertex buffer are, setup frame buffer

Confidential

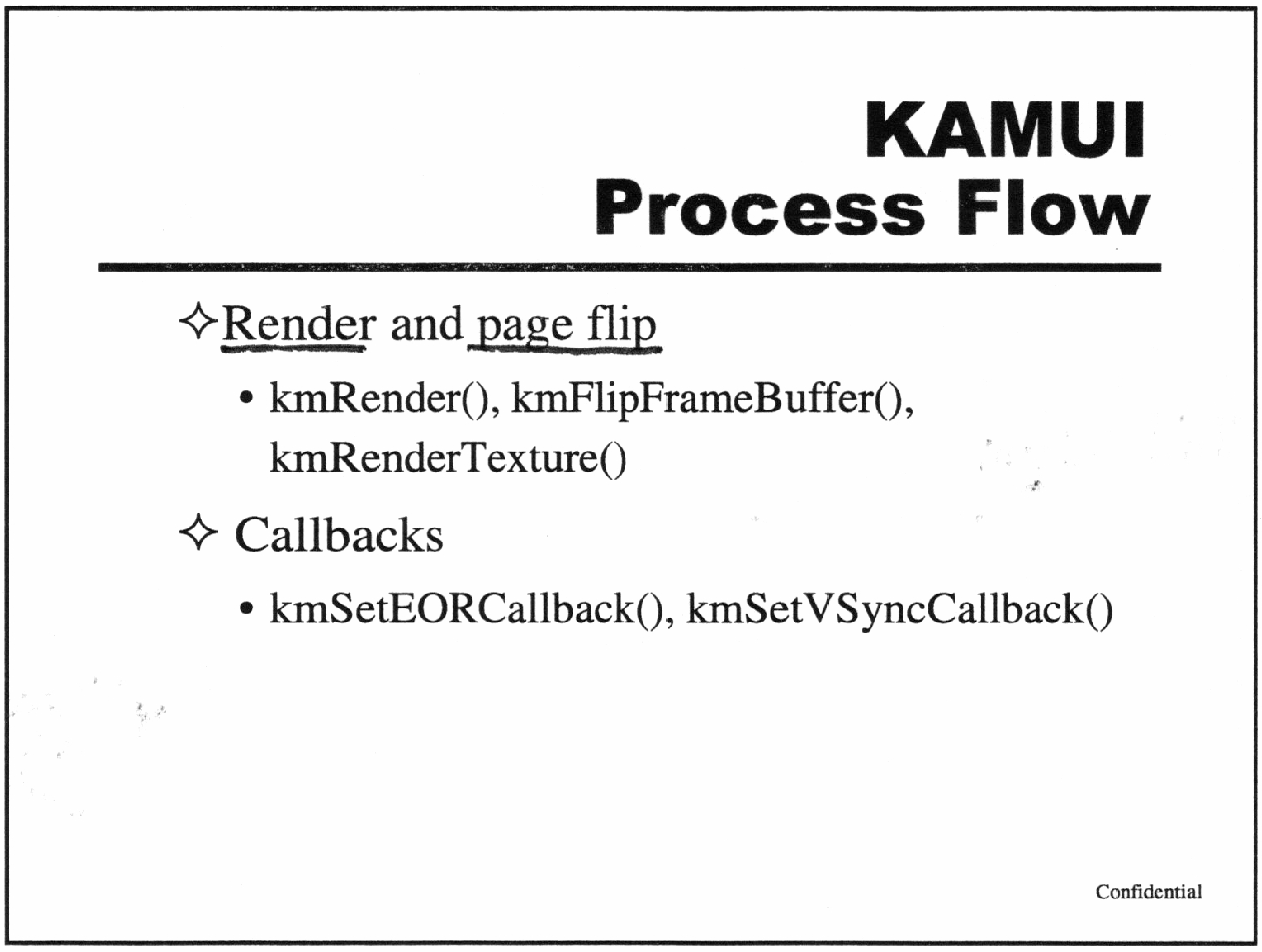


KAMUI Process Flow

- ✧ Global scene parameters
  - kmSetFogTableColor(), kmSetUserClipping()
- ✧ Texture loading
  - kmLoadTexture(), kmQueryFinishLastTextureDMA()
- ✧ Vertex and control parameter registration
  - kmSetVertex(), kmSetVertexRenderState()

Confidential

Things Can't you
can't change.
while rendering



KAMUI Process Flow

- ✧ Render and page flip
  - kmRender(), kmFlipFrameBuffer(), kmRenderTexture()
- ✧ Callbacks
  - kmSetEORCallback(), kmSetVSyncCallback()

Confidential

KAMUI Display Modes

Display Generator

- ✧ VGA / NTSC / PAL
- ✧ Display modes
  - 320x240, 320x480, 640x240, 640x480
  - Interlace (30 Hz), non-interlace (60 Hz), pseudo-NI (60 Hz fields), flicker-free (60 Hz averaged)

Confidential

KAMUI Display Modes

Frame Buffer

Most people

- ✧ Color depth
  - 16-bit (RGB 565 / 555), 24-bit (RGB 888)
- ✧ Render options
  - 16-bit dither
  - Antialiasing filter (4X scene render and scale)

Can be turned on on a per frame basas

Confidential

KAMUI Scene Parameters

Global Scene Render State

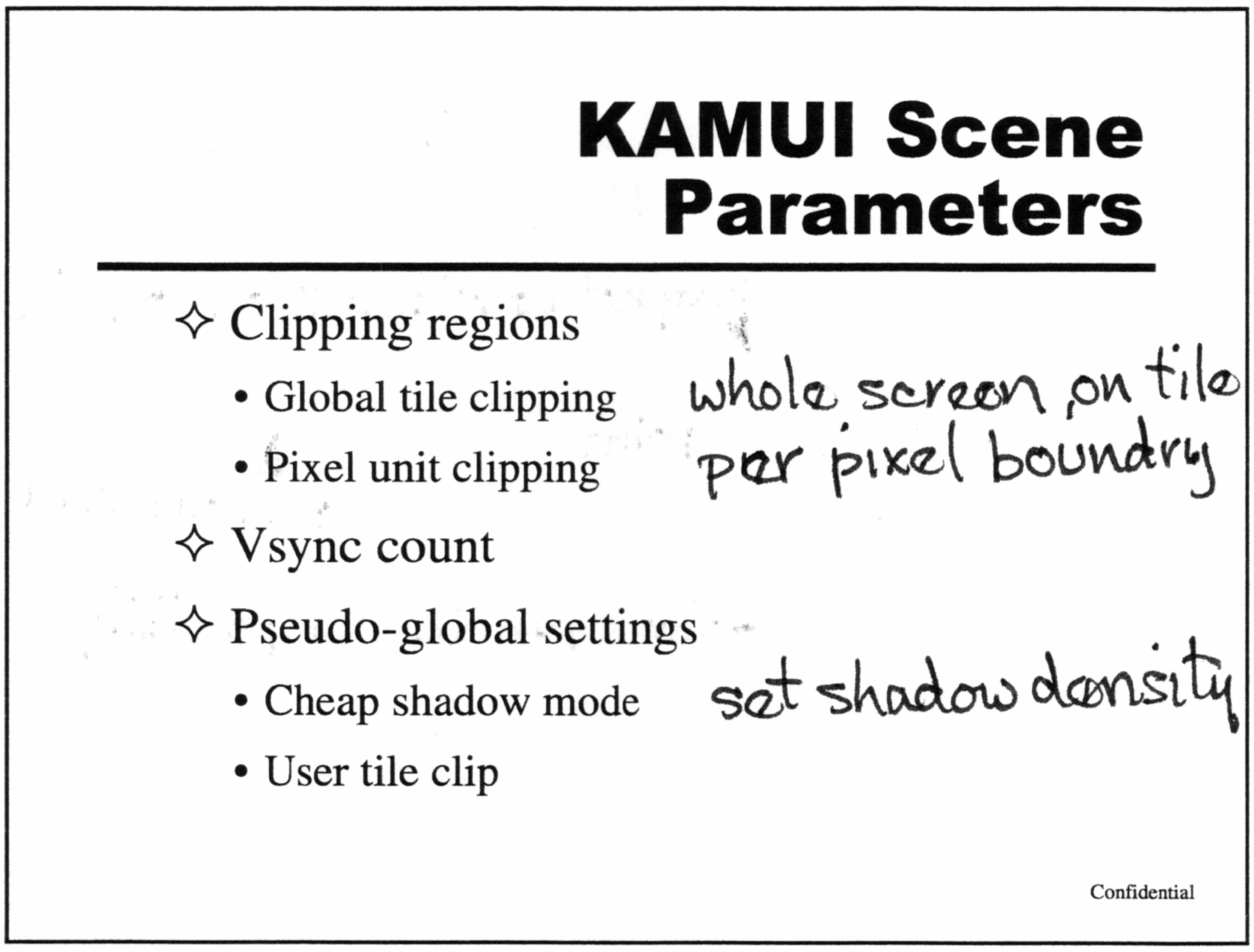
- ✧ Culling register
- ✧ Color clamp (min / max)
- ✧ Fog settings
  - ① • Vertex/table fog color, fog density
  - ② • Set fog table

Confidential

KAMUI Scene Parameters

- ✧ Global palette settings
  - Palette mode/bit-depth
  - Palette table
- ✧ Border color ◄ for debuging
- ✧ Translucent autosort mode
- ✧ Texture stride width

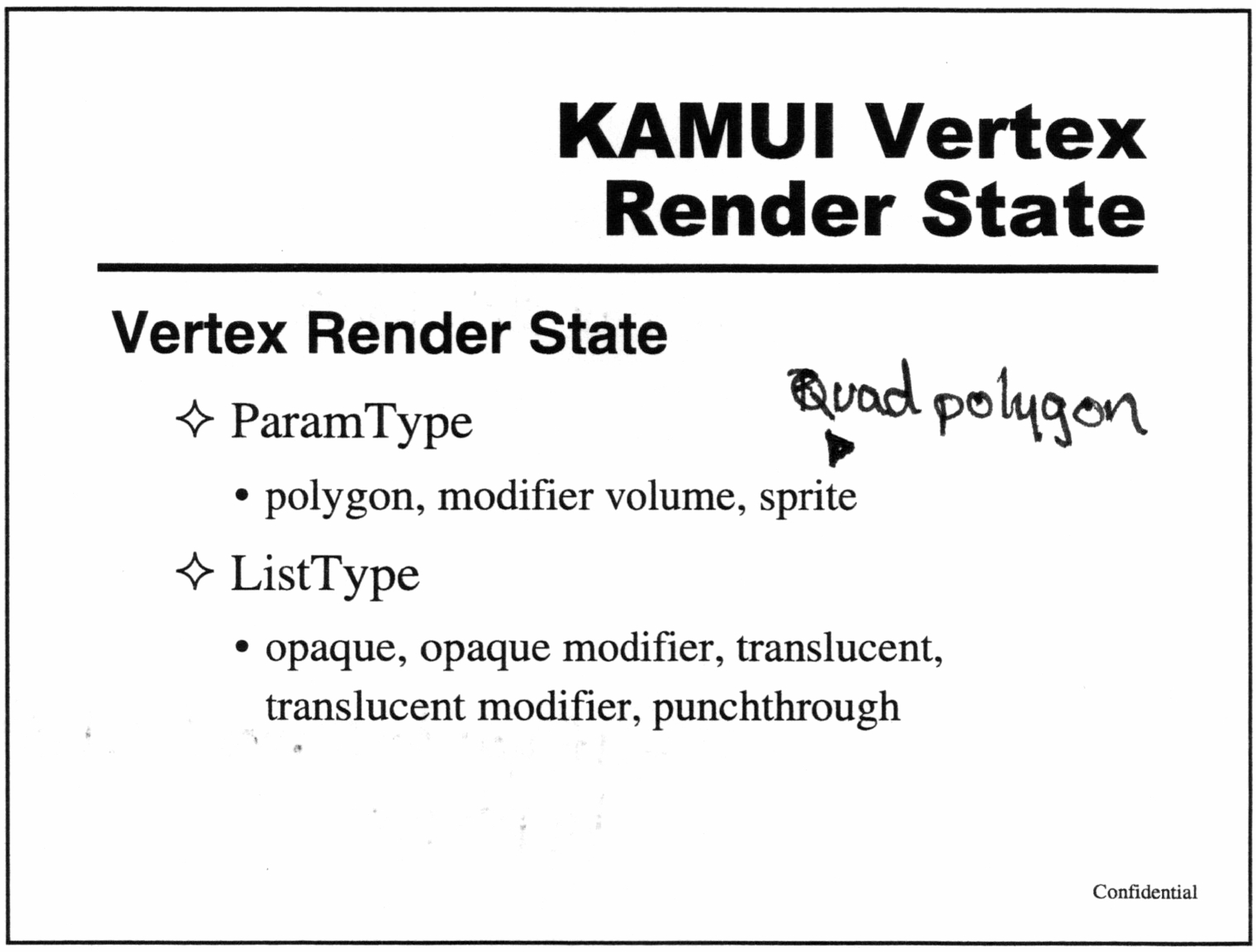
Confidential



KAMUI Scene Parameters

- ✧ Clipping regions
  - Global tile clipping whole screen on tile boundries
  - Pixel unit clipping per pixel boundry
- ✧ Vsync count
- ✧ Pseudo-global settings
  - Cheap shadow mode set shadow density
  - User tile clip

Confidential



KAMUI Vertex Render State

Vertex Render State

- ✧ ParamType Quad polygon
  - polygon, modifier volume, sprite
- ✧ ListType
  - opaque, opaque modifier, translucent, translucent modifier, punchthrough

Confidential

KAMUI Vertex Render State

- ✧ ColorType May not be effishant on SH4 side
  - packed ARGB, floating, intensity, prev.intensity

    point

    All polys use base intensity
- ✧ UVFormat
  - packed 16-bit, floating-point
- ✧ DepthMode - for OpenGL compatibility
  - ignore, <, <=, =, !=, >=, >, always

Confidential

KAMUI Vertex Render State

- ✧ CullingMode — per strip basis
  - small polygons, CW, CCW, none (CW/CCW also cull small polygons)
- ✧ ShadingMode
  - flat shaded, Gouraud, flat shaded & textured, Gouraud & textured
- ✧ bZwriteDisable - If you don't want z buffer updated by polygon.

Confidential

KAMUI Vertex Render State

- ✧ SRCBlendingMode, DSTBlendingMode
  - zero, one, src alpha/color, dest alpha/color, inv src alpha/color, inv dest alpha/color, both src alpha, both inv src alpha
- ✧ FogMode
  - table, table 2 (color & alpha), vertex, no fog

Applied to all vertexs

Specified in Offset color Alpha value

- ✧ bUseSpecular

Confidential

KAMUI Vertex Render State

- ✧ bUseAlpha, bIgnoreTextureAlpha
- ✧ ClampUV, FlipUV
  - U, V, both, none
- ✧ FilterMode
  - point sample, bilinear, trilinear (1 or 2 pass)

Use (pracitly free)

Use to avoid hard mip mapping "poping"

- ✧ bSuperSample [anisotropic filter]

five surrounding pixels in an oval

Confidential

KAMUI Vertex Render State

- ✧ MipMapAdjust (0.25 - 3.75, default 1.0)
- ✧ TextureShadingMode
  - decal (tex + offset), decal alpha,
    Mainly → modulate (tex * shade + offset), modulate alpha
- ✧ bColorClamp
- ✧ PaletteBank (0 - 63, 8-bit = 0, 16, 32, 48)

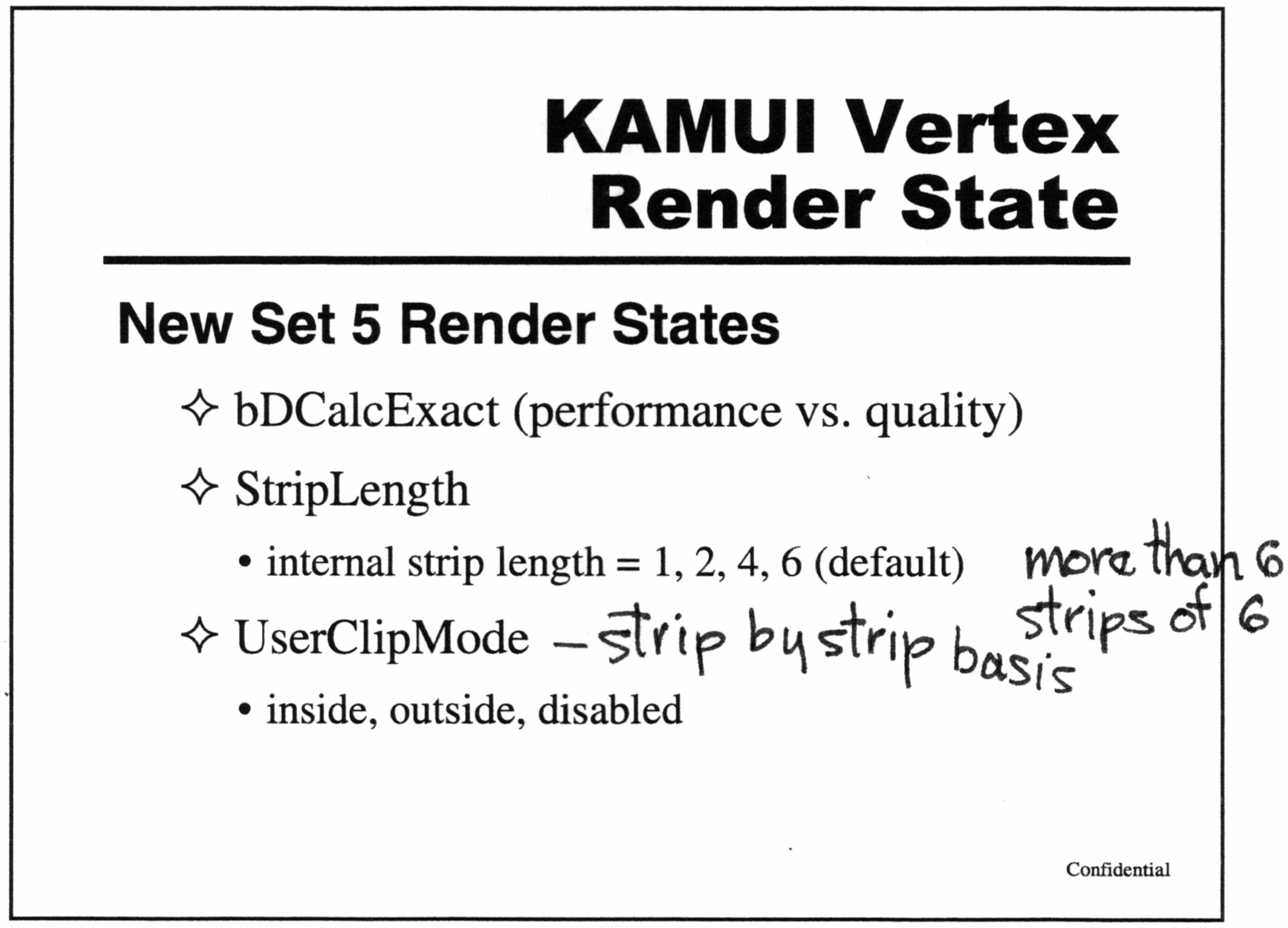
Supe pal 1024
4 banks

Confidential

KAMUI Vertex Render State

- ✧ fFaceColor[4]
  - alpha, R, G, B, specified for intesity vertex type
- ✧ fOffsetColor[4]
  - alpha, R, G, B, for specular highlight intensity
- ✧ fBoundingBox[4] — only for modifier volume
  - xmin, xmax, ymin, ymax, bounds modifer volume

Confidential

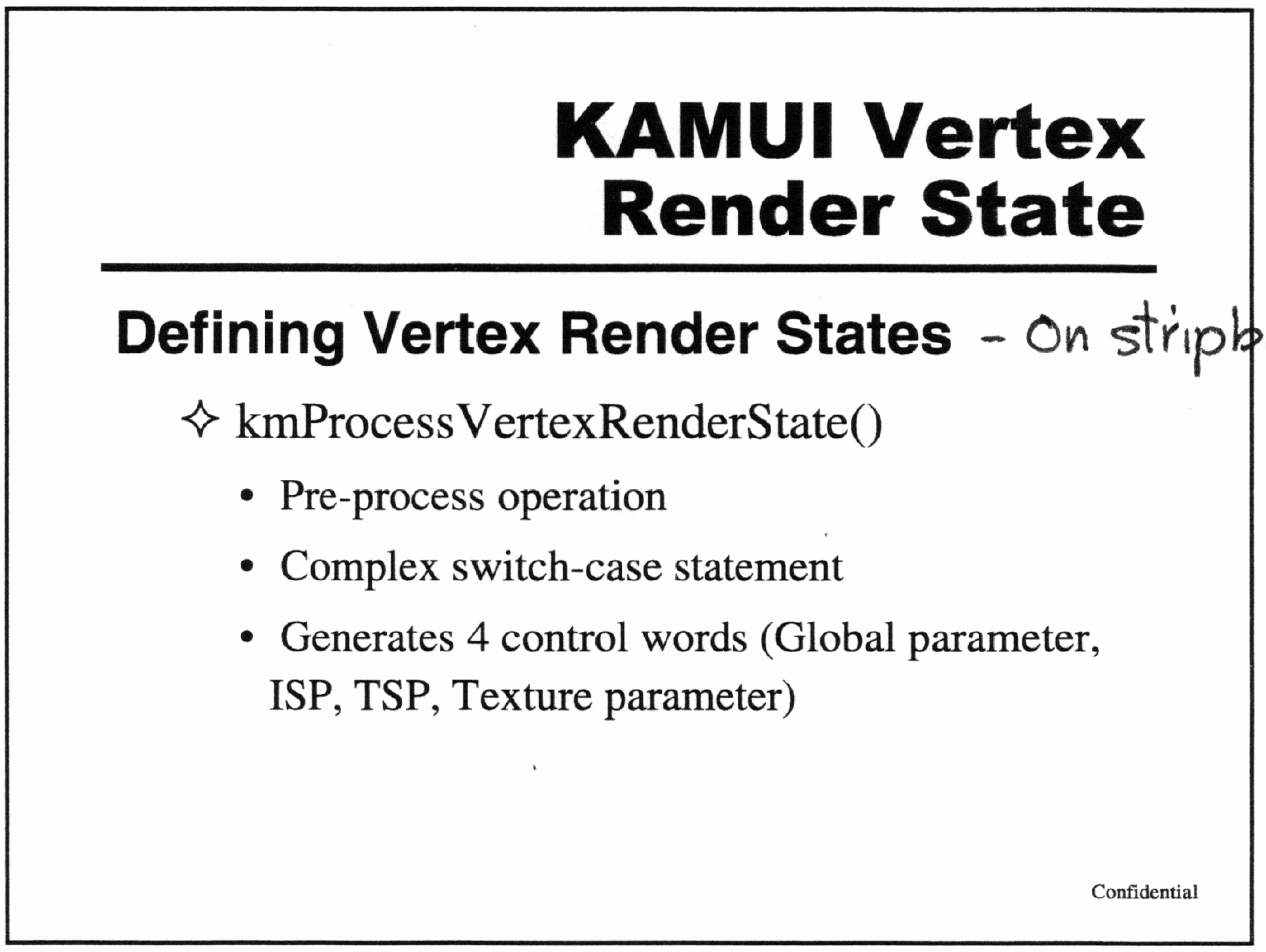


KAMUI Vertex Render State

New Set 5 Render States

- bDCalcExact (performance vs. quality)
- StripLength
  - internal strip length = 1, 2, 4, 6 (default) more than 6 - chopped into strips of 6
- UserClipMode – strip by strip basis
  - inside, outside, disabled

Confidential



KAMUI Vertex Render State

Defining Vertex Render States - on strip by strip basis

- kmProcessVertexRenderState()
  - Pre-process operation
  - Complex switch-case statement
  - Generates 4 control words (Global parameter, ISP, TSP, Texture parameter)

Confidential

KAMUI Vertex Render State

Switching Vertex Render States

✧ kmSetVertexRenderState()

- Switches state of global KAMUI render context

✧ kmStartVertexStrip()

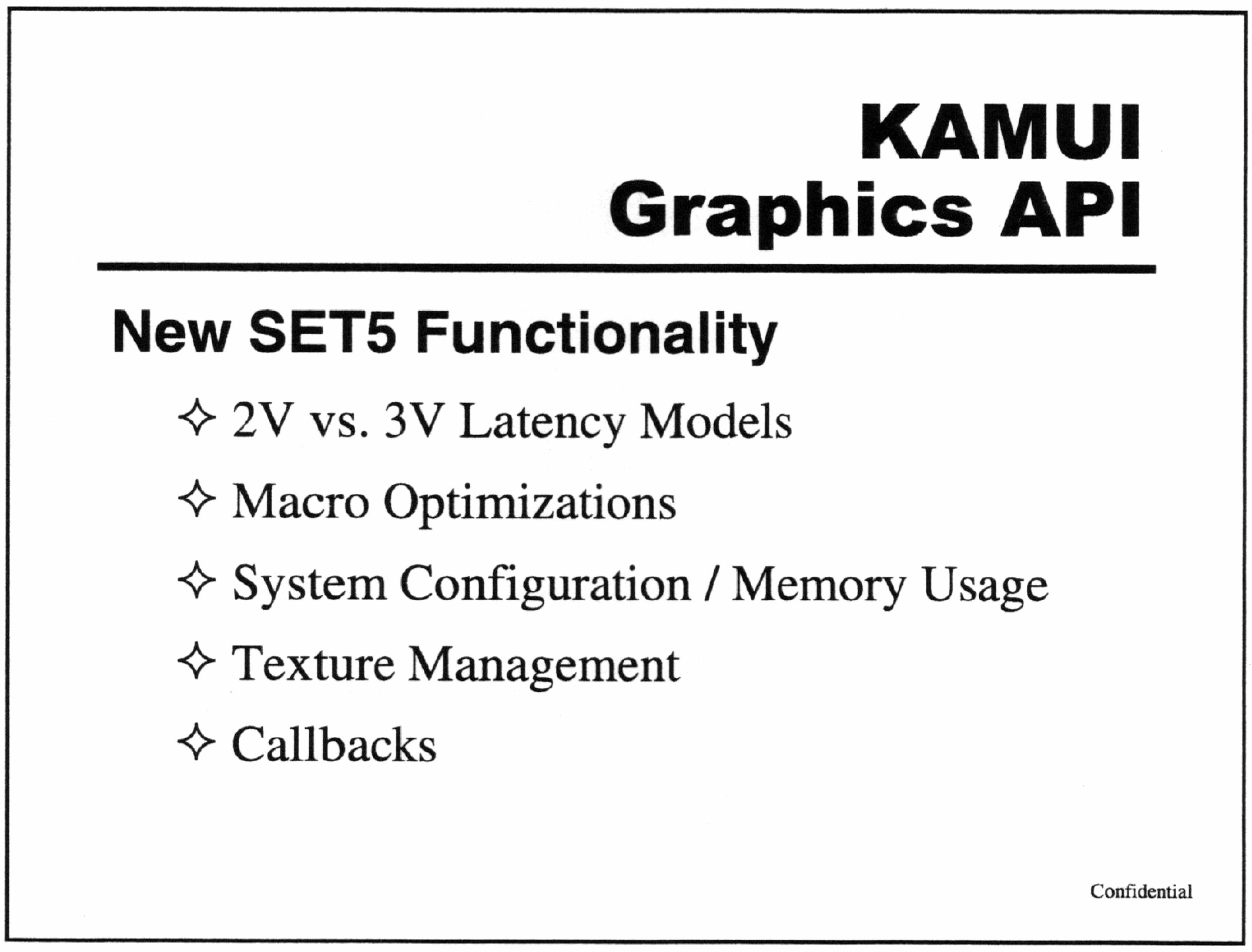
- Apply vertex render state to new strip
- Global parameter inserted in vertex buffer

Confidential

KAMUI Graphics API

New SET5 Functionality

Confidential

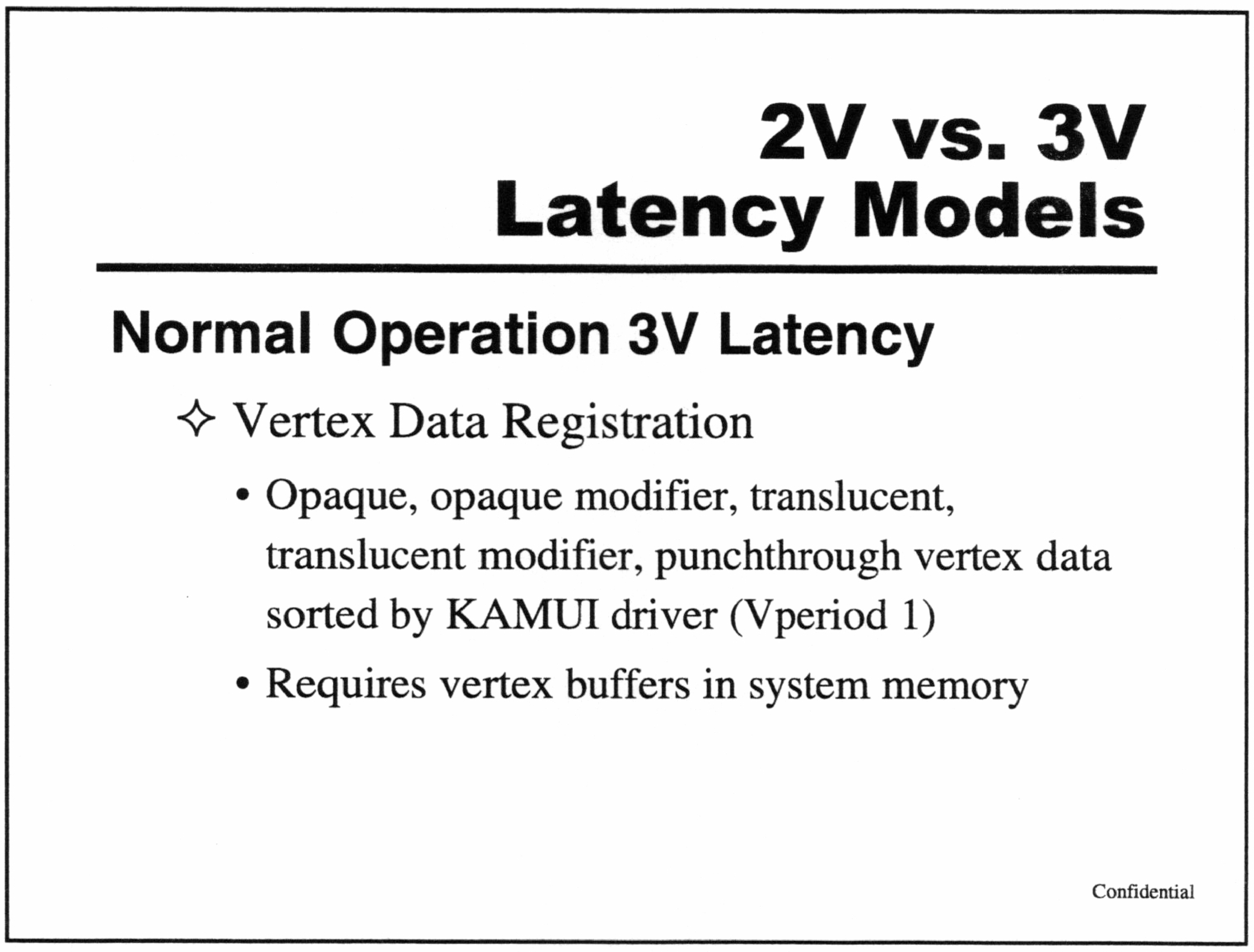


KAMUI
Graphics API

New SET5 Functionality

- ✧ 2V vs. 3V Latency Models
- ✧ Macro Optimizations
- ✧ System Configuration / Memory Usage
- ✧ Texture Management
- ✧ Callbacks

Confidential



2V vs. 3V
Latency Models

Normal Operation 3V Latency

- ✧ Vertex Data Registration
  - Opaque, opaque modifier, translucent, translucent modifier, punchthrough vertex data sorted by KAMUI driver (Vperiod 1)
  - Requires vertex buffers in system memory

Confidential

2V vs. 3V Latency Models

✧ 3V Latency Model

- Transfer of vertex data to TA native buffer takes advantage of high speed burst mode DMA transfer (Vperiod 2)
- Scene is rendered (Vperiod 3), Total latency 3V
- Yields potentially slow user input response time, 1V for joystick input + 3V = 4V response

Confidential

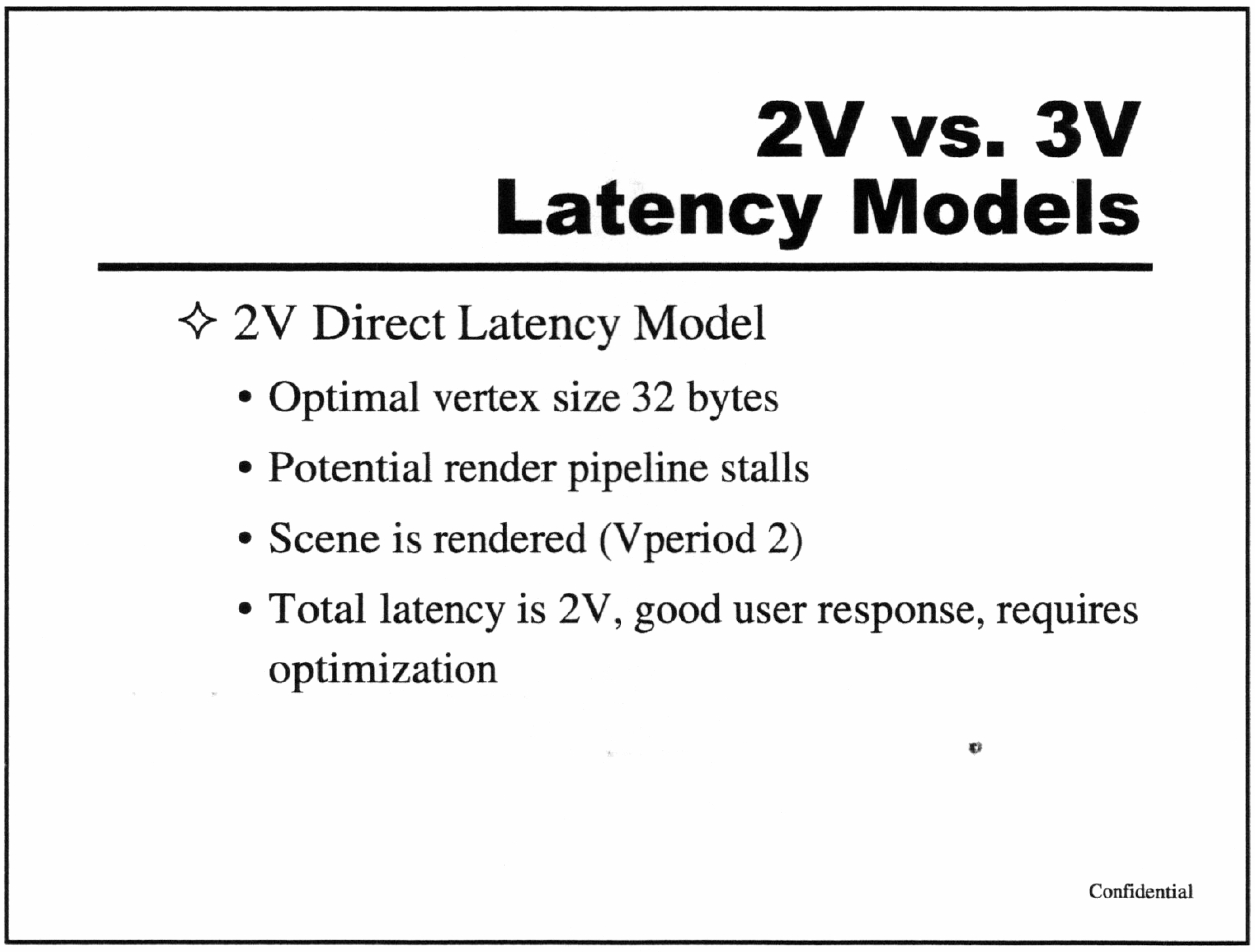
2V vs. 3V Latency Models

Direct Mode 2V Latency

✧ Vertex Data Registration

- Vertex data and control parameters passed directly to TA using Store Queue (Vperiod 1)
- Data must be sent in sorted order
- No system memory required

Confidential

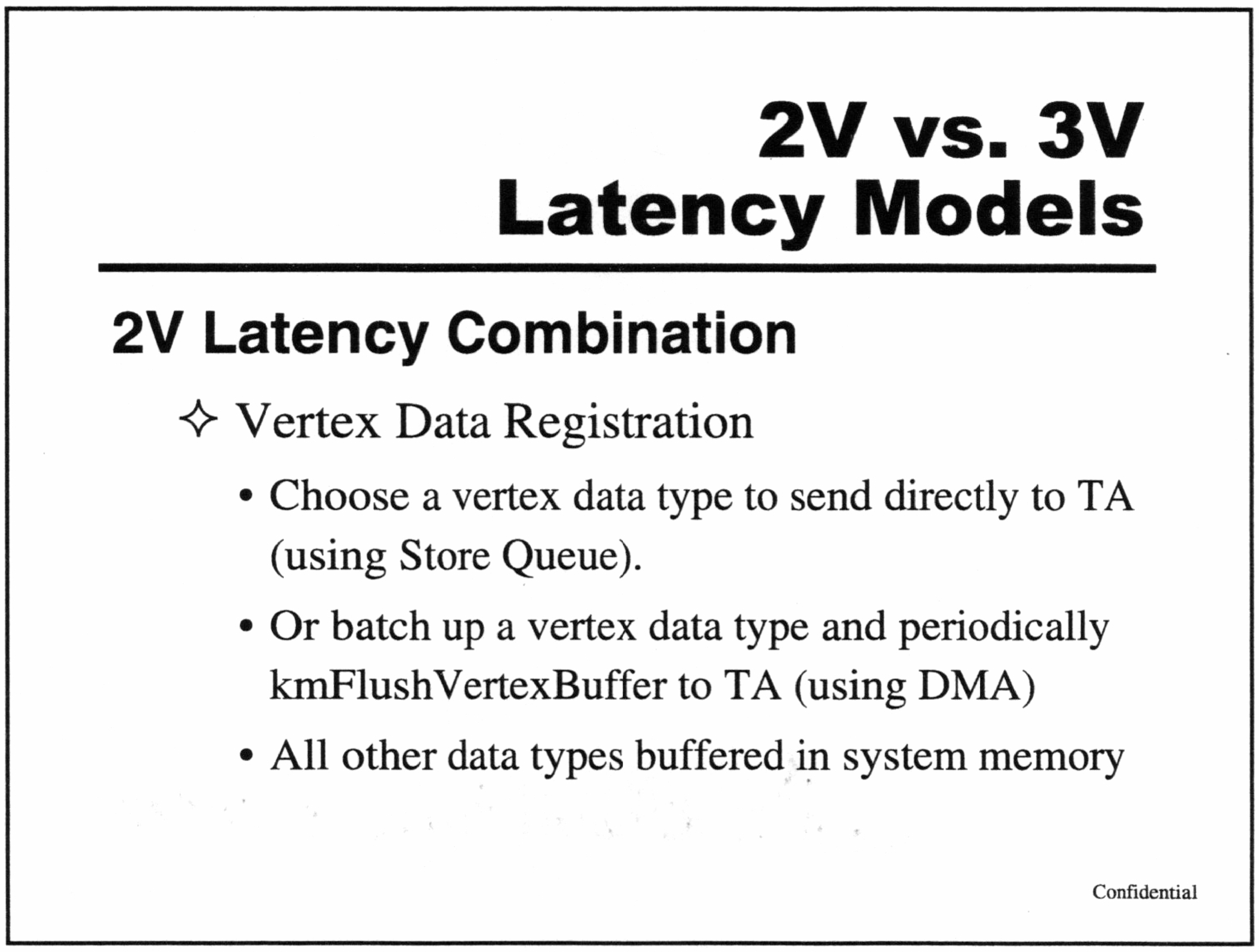


2V vs. 3V Latency Models

✧ 2V Direct Latency Model

- Optimal vertex size 32 bytes
- Potential render pipeline stalls
- Scene is rendered (Vperiod 2)
- Total latency is 2V, good user response, requires optimization

Confidential



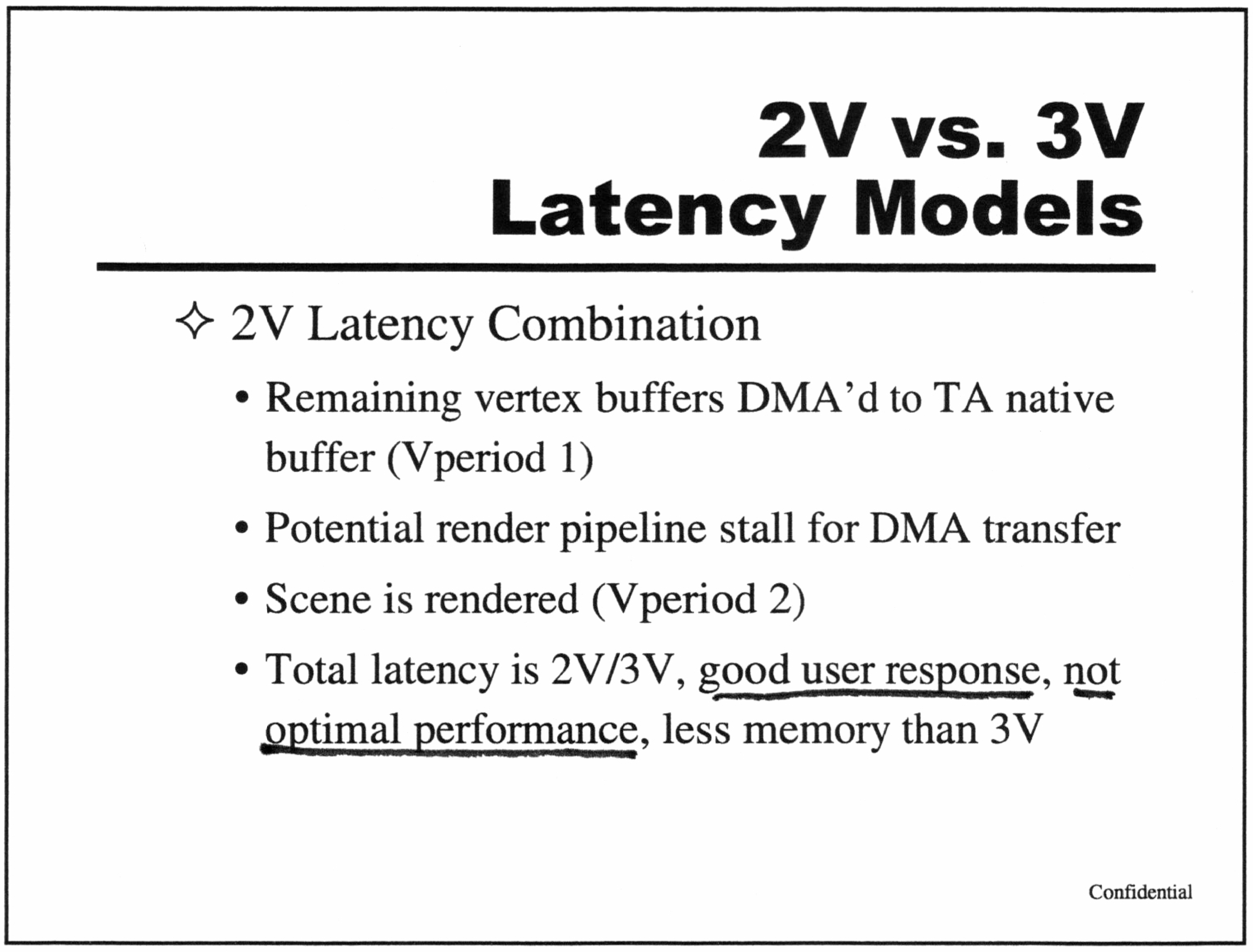
2V vs. 3V Latency Models

2V Latency Combination

✧ Vertex Data Registration

- Choose a vertex data type to send directly to TA (using Store Queue).
- Or batch up a vertex data type and periodically kmFlushVertexBuffer to TA (using DMA)
- All other data types buffered in system memory

Confidential

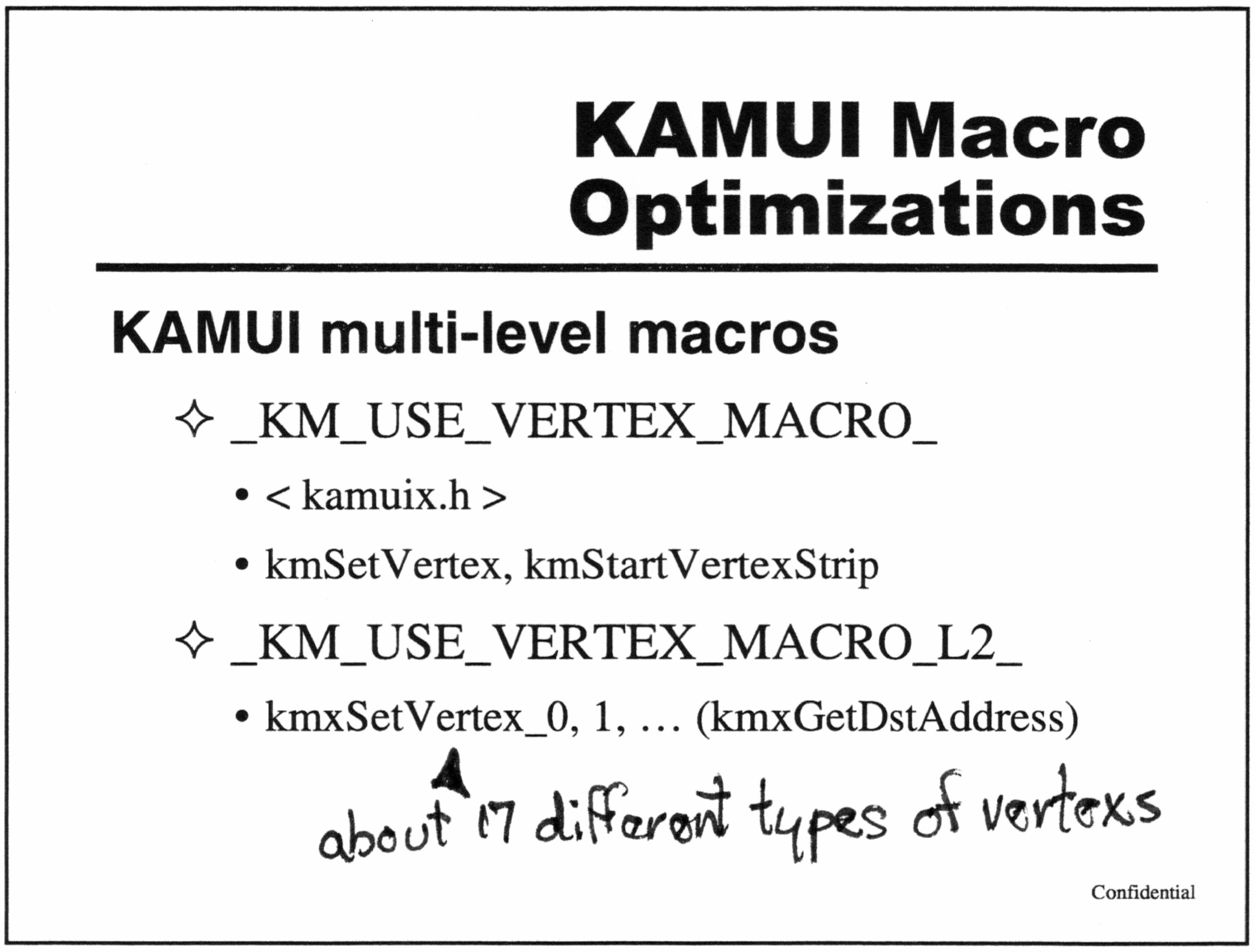


2V vs. 3V Latency Models

✧ 2V Latency Combination

- Remaining vertex buffers DMA'd to TA native buffer (Vperiod 1)
- Potential render pipeline stall for DMA transfer
- Scene is rendered (Vperiod 2)
- Total latency is 2V/3V, good user response, not optimal performance, less memory than 3V

Confidential



KAMUI Macro Optimizations

KAMUI multi-level macros

✧ _KM_USE_VERTEX_MACRO_

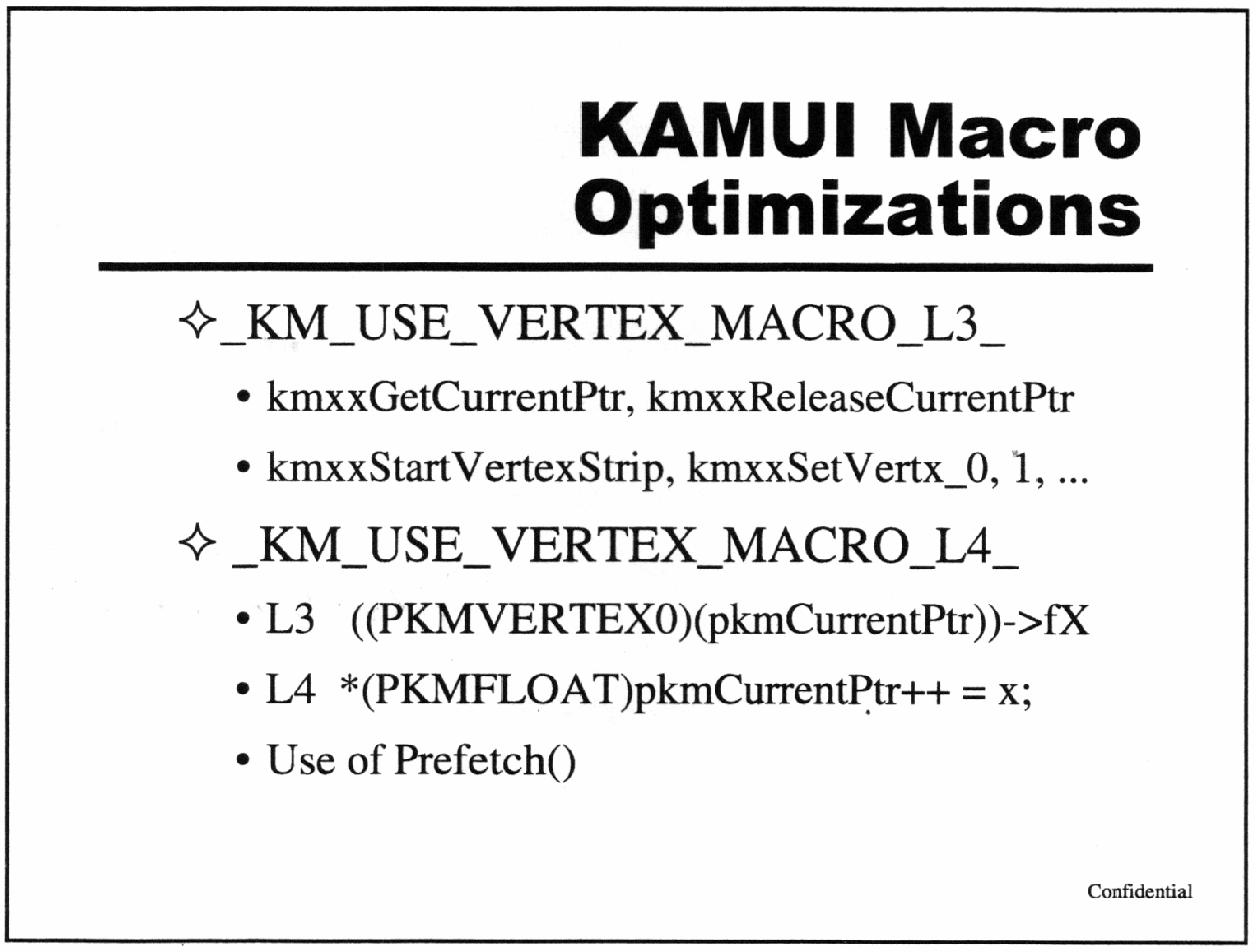
- < kamuix.h >
- kmSetVertex, kmStartVertexStrip

✧ _KM_USE_VERTEX_MACRO_L2_

- kmxSetVertex_0, 1, … (kmxGetDstAddress)

about 17 different types of vertexs

Confidential



KAMUI Macro Optimizations

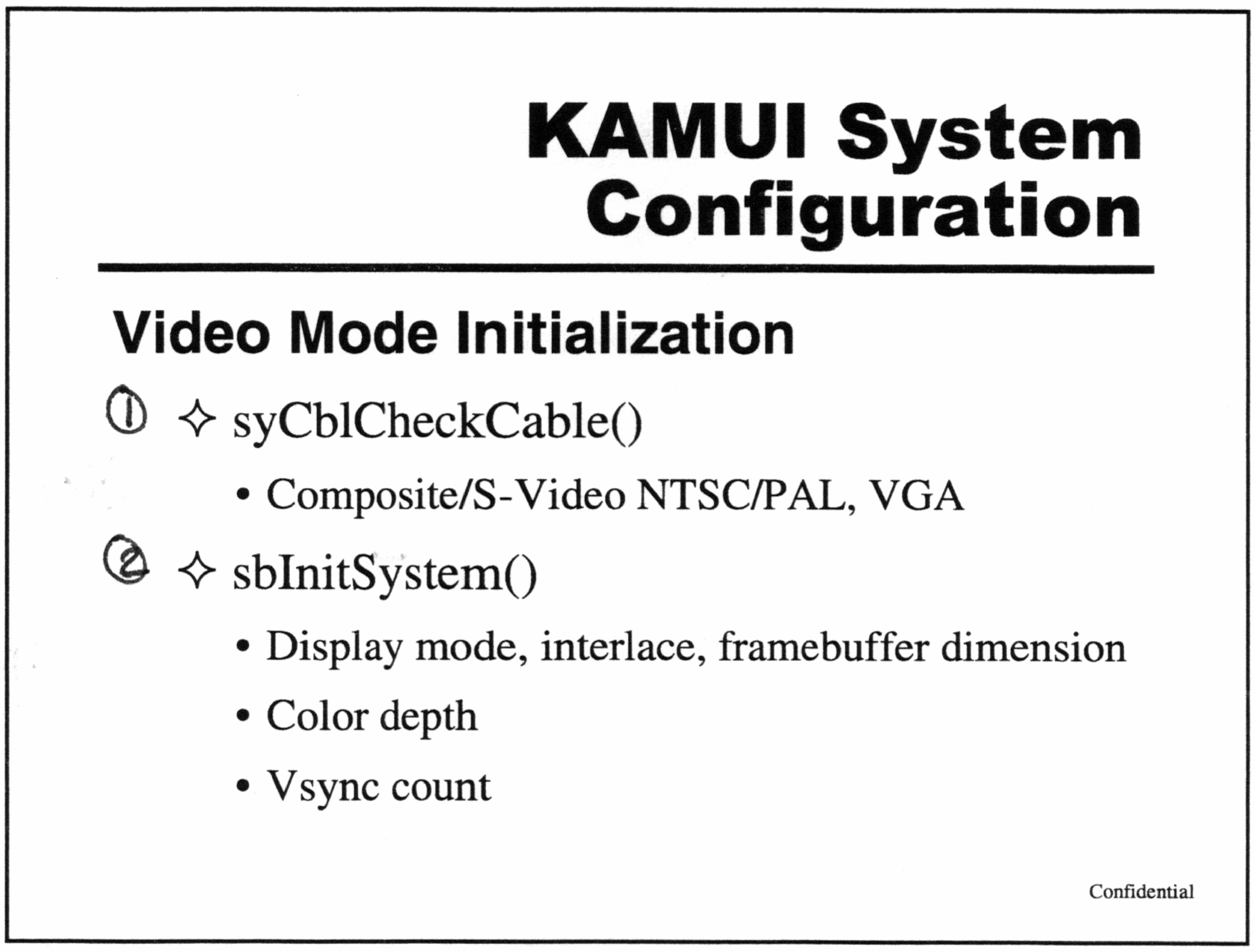
✧_KM_USE_VERTEX_MACRO_L3_

- kmxxGetCurrentPtr, kmxxReleaseCurrentPtr
- kmxxStartVertexStrip, kmxxSetVertx_0, 1, ...

✧ _KM_USE_VERTEX_MACRO_L4_

- L3 ((PKMVERTEX0)(pkmCurrentPtr))->fX
- L4 *(PKMFLOAT)pkmCurrentPtr++ = x;
- Use of Prefetch()

Confidential



KAMUI System Configuration

Video Mode Initialization

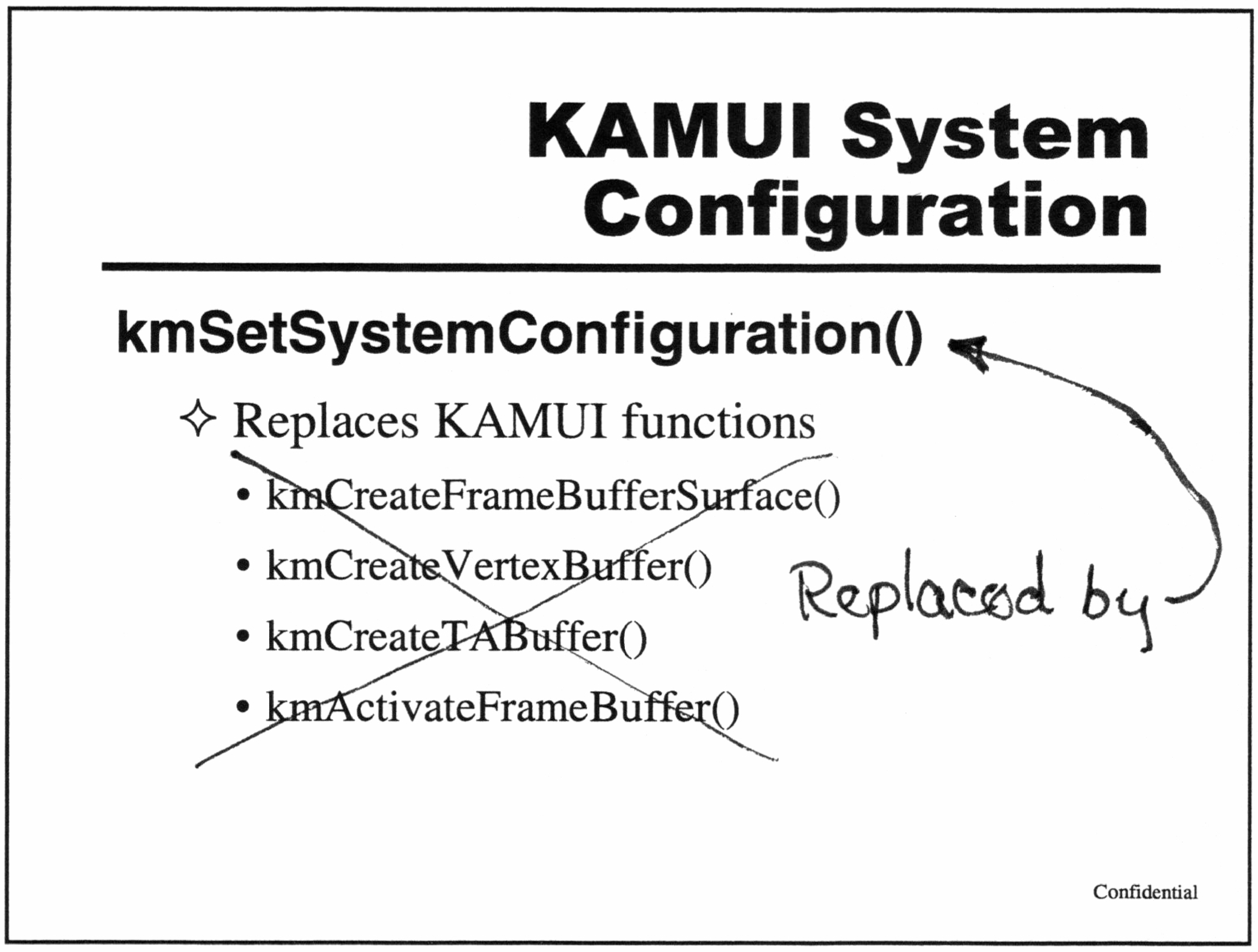
① ✧ syCblCheckCable()

- Composite/S-Video NTSC/PAL, VGA

② ✧ sbInitSystem()

- Display mode, interlace, framebuffer dimension
- Color depth
- Vsync count

Confidential



KAMUI System Configuration

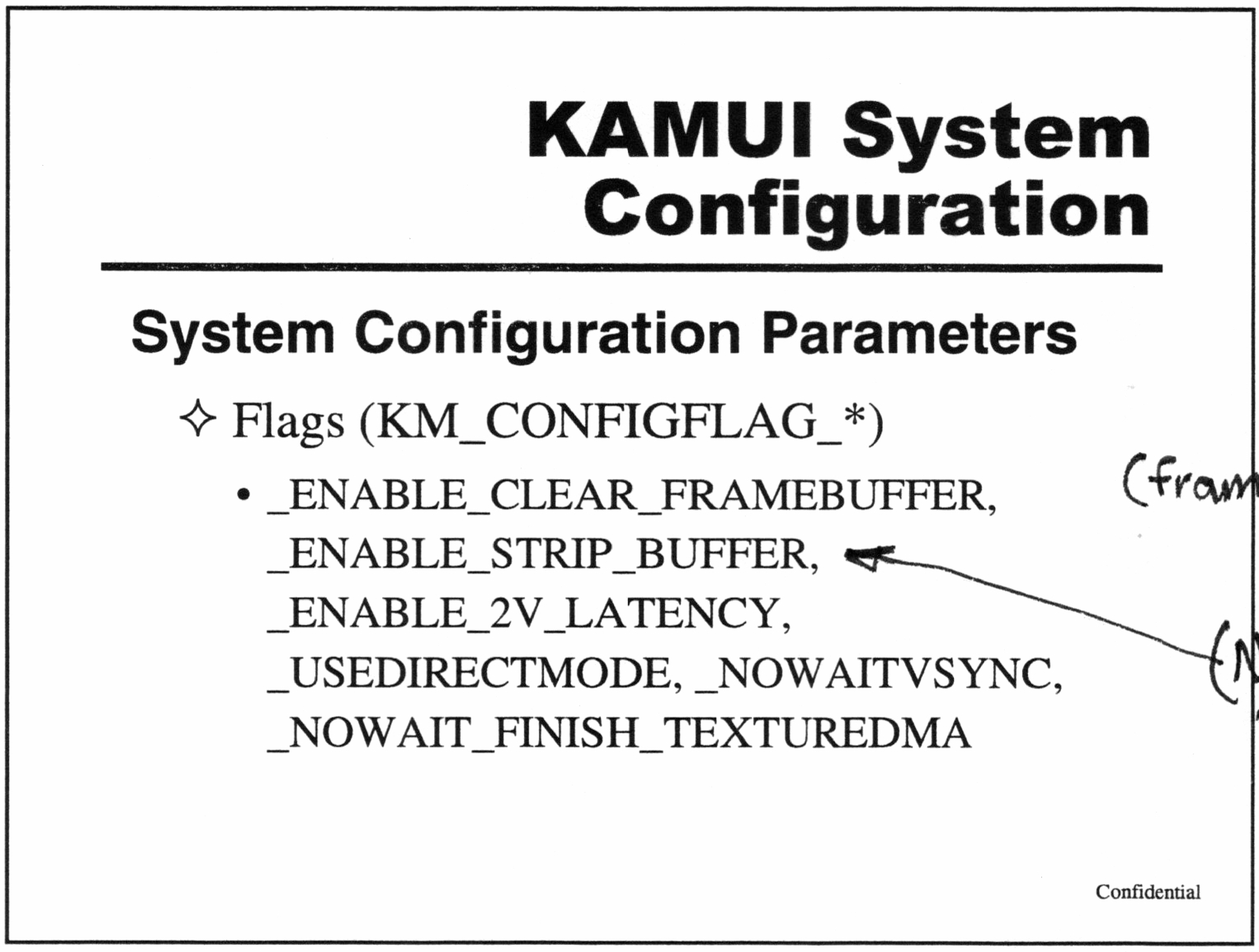
kmSetSystemConfiguration()

✧ Replaces KAMUI functions

- ~~kmCreateFrameBufferSurface()~~
- ~~kmCreateVertexBuffer()~~
- ~~kmCreateTABuffer()~~
- ~~kmActivateFrameBuffer()~~

Replaced by

Confidential



KAMUI System Configuration

System Configuration Parameters

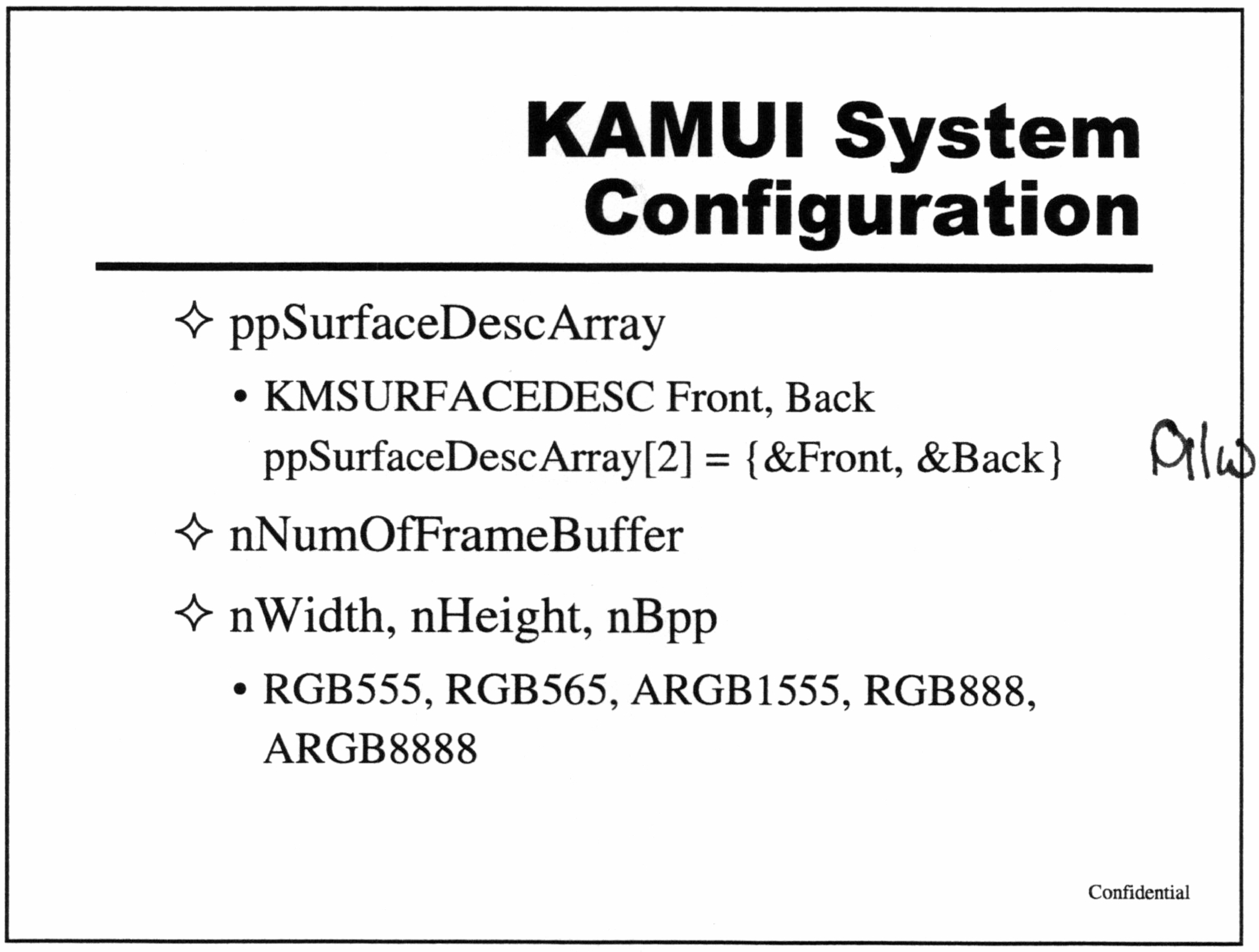
✧ Flags (KM_CONFIGFLAG_*)

- _ENABLE_CLEAR_FRAMEBUFFER, _ENABLE_STRIP_BUFFER, _ENABLE_2V_LATENCY, _USEDIRECTMODE, _NOWAITVSYNC, _NOWAIT_FINISH_TEXTUREDMA

(frame buffer always cleared)

(No frame buffer needed if 60 Hz)

Confidential

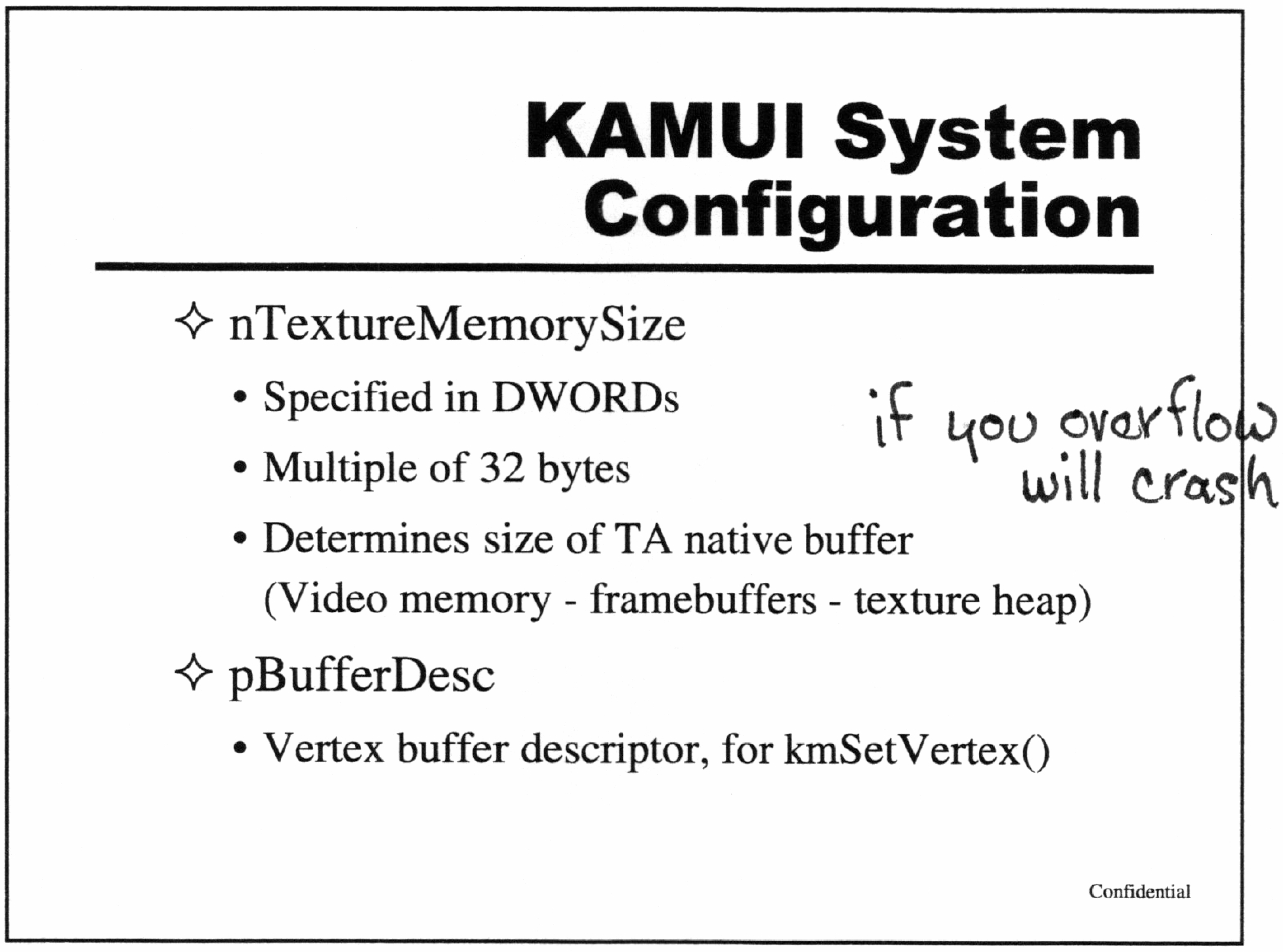


KAMUI System Configuration

- ✧ ppSurfaceDescArray
  - KMSURFACEDESC Front, Back
    ppSurfaceDescArray[2] = {&Front, &Back}
- ✧ nNumOfFrameBuffer
- ✧ nWidth, nHeight, nBpp
  - RGB555, RGB565, ARGB1555, RGB888, ARGB8888

Confidential

Always 2 (2 frame buffers)

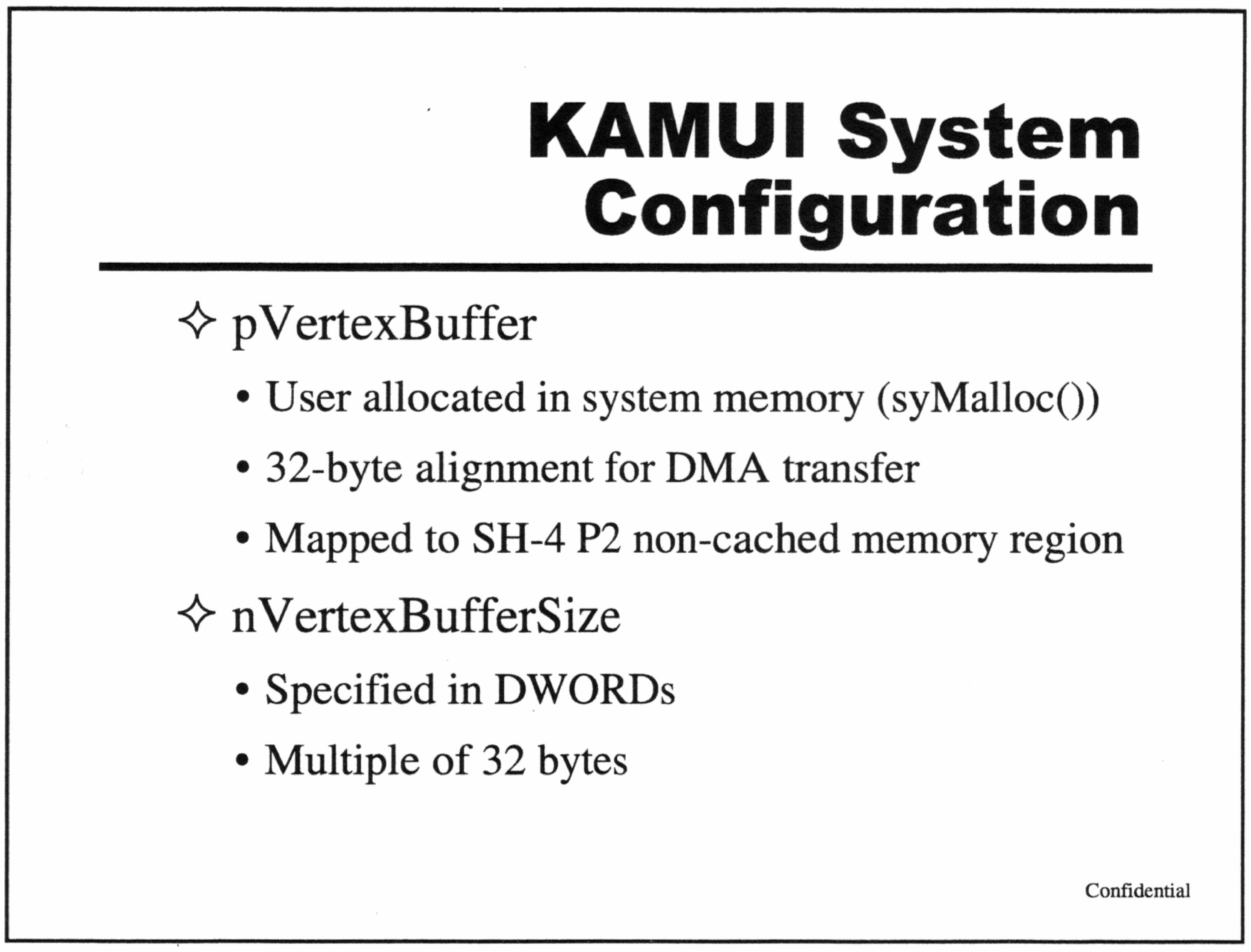


KAMUI System Configuration

- ✧ nTextureMemorySize
  - Specified in DWORDs
  - Multiple of 32 bytes
  - Determines size of TA native buffer
    (Video memory - framebuffers - texture heap)
- ✧ pBufferDesc
  - Vertex buffer descriptor, for kmSetVertex()

Confidential

if you overflow, the game will crash.



KAMUI System Configuration

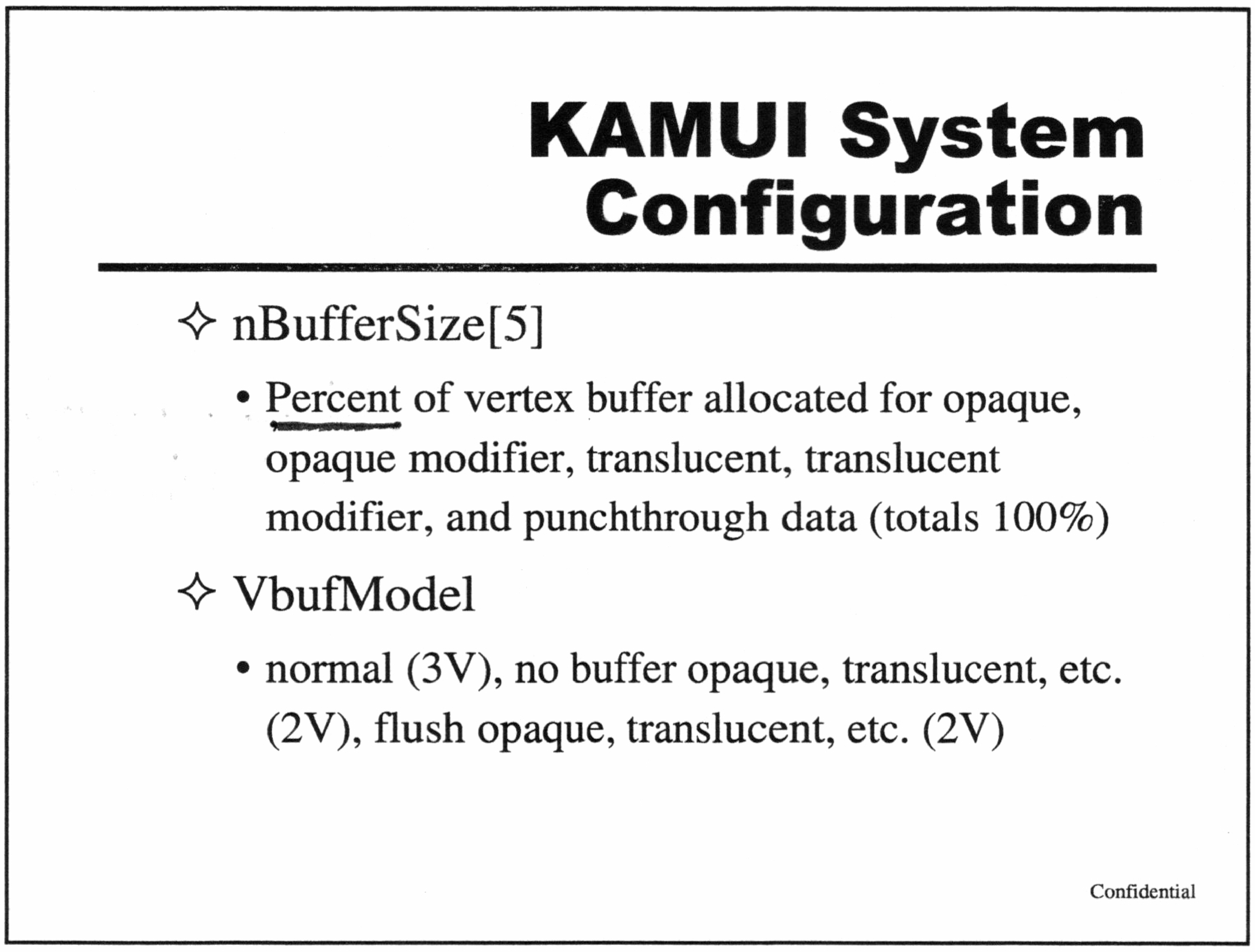
✧ pVertexBuffer

- User allocated in system memory (syMalloc())
- 32-byte alignment for DMA transfer
- Mapped to SH-4 P2 non-cached memory region

✧ nVertexBufferSize

- Specified in DWORDs
- Multiple of 32 bytes

Confidential



KAMUI System Configuration

✧ nBufferSize[5]

- Percent of vertex buffer allocated for opaque, opaque modifier, translucent, translucent modifier, and punchthrough data (totals 100%)

✧ VbufModel

- normal (3V), no buffer opaque, translucent, etc. (2V), flush opaque, translucent, etc. (2V)

Confidential

KAMUI System Configuration

Direct Mode Considerations

✧ SystemConfiguration

- KM_CONFIGFLAG_USEDIRECTMODE
- VertexBufferDesc.fActiveList = KM_ACTIVE_OPAQUE_POLYGON | TRANS_POLYGON, etc.

Confidential

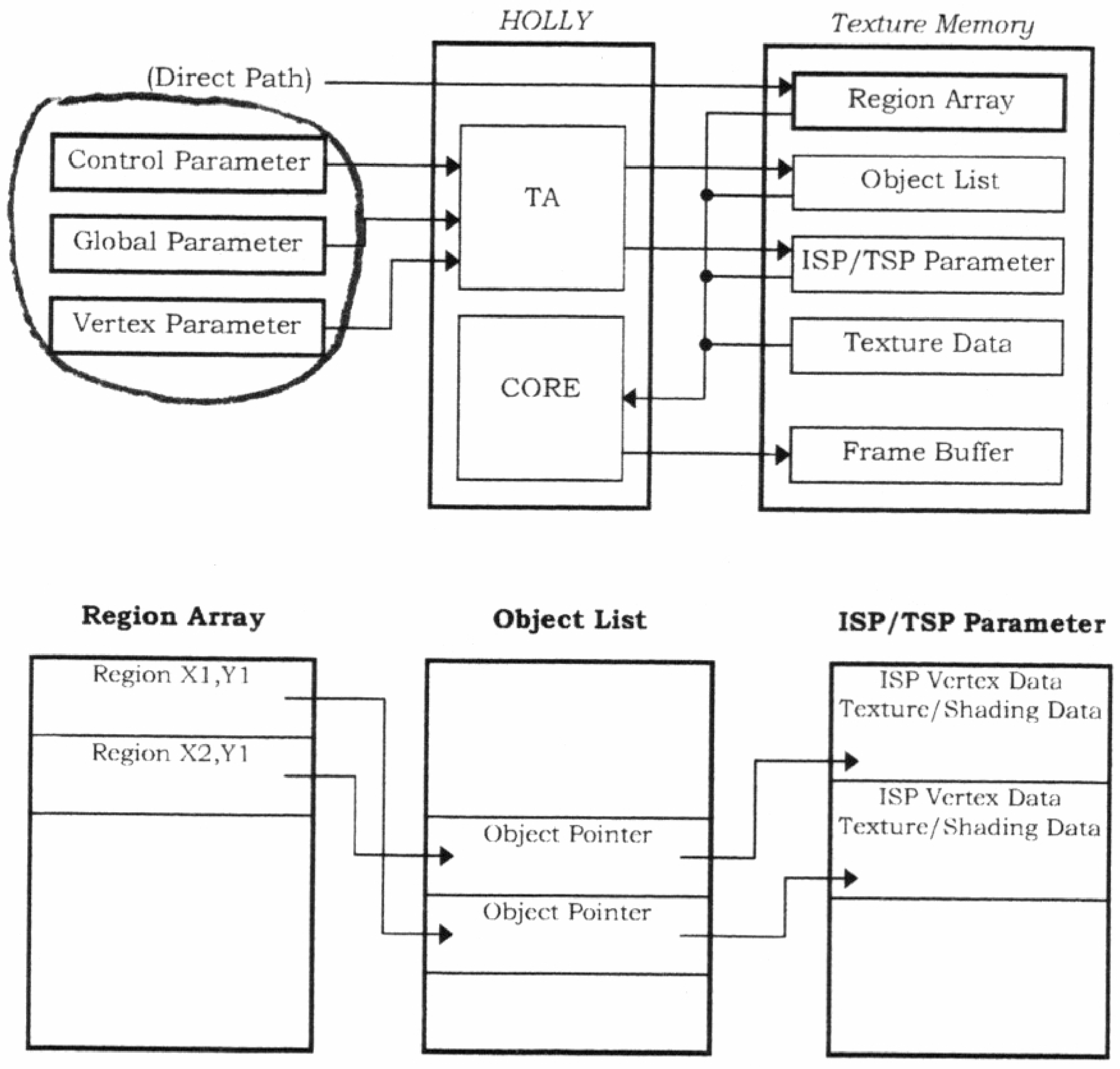
KAMUI System Configuration

✧ Direct-mode functions

- kmStartVertexStripDirect()
- kmSetVertexDirect()
- kmSetEndOfListDirect() — tells it this is the end of OPA or
- kmSetUserClippingDirect()
- kmRenderDirect()

Confidential

Display List Details



KAMUI Memory Configuration

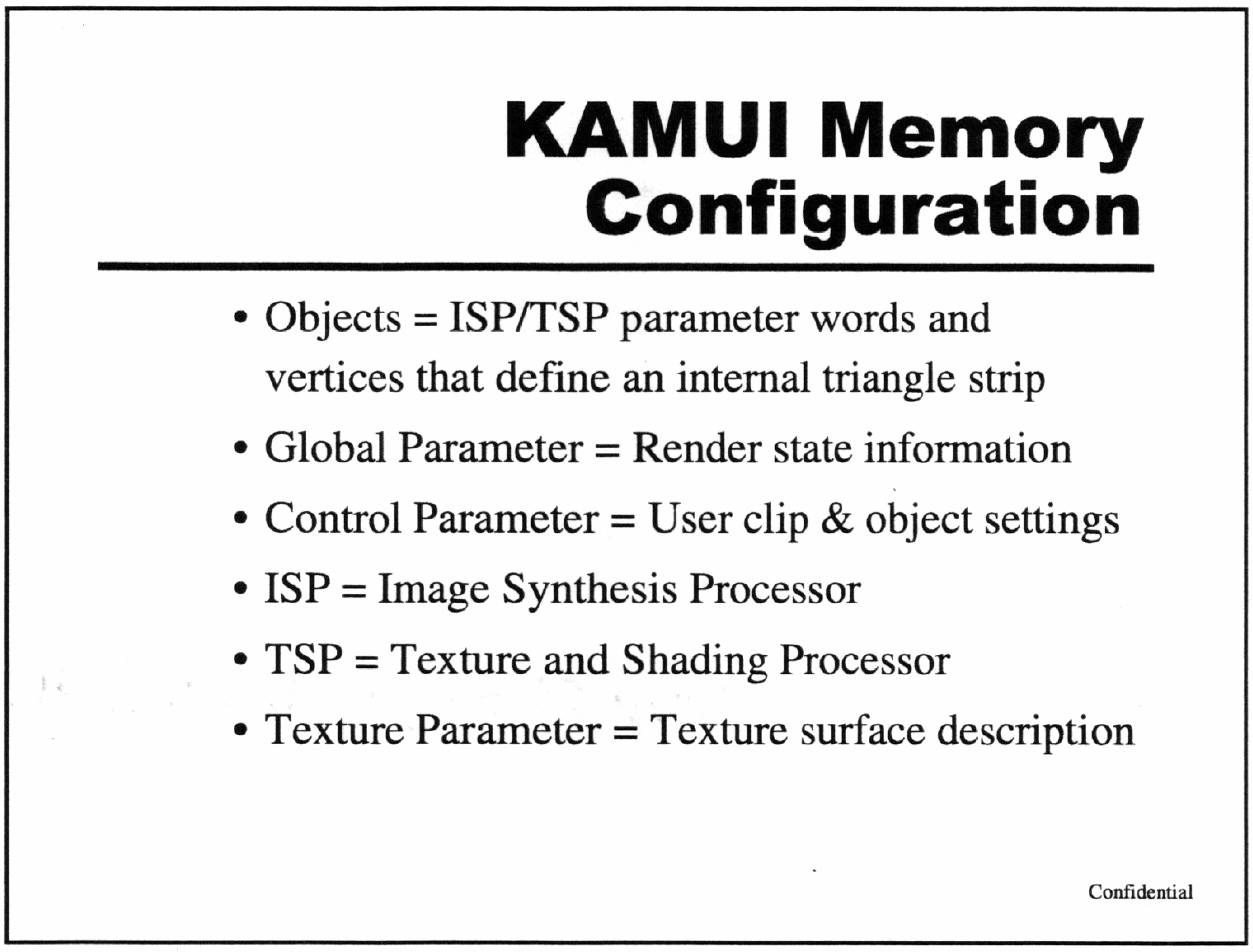
Definition of Terms

- Region Array = Array of 32x32 tiles, containing head Object Pointers for each of the 5 list types

PSydo

- Object Pointer List = Linked lists of pointers to objects 'potentially' in each tile (generated by TA bounding box algorithm)

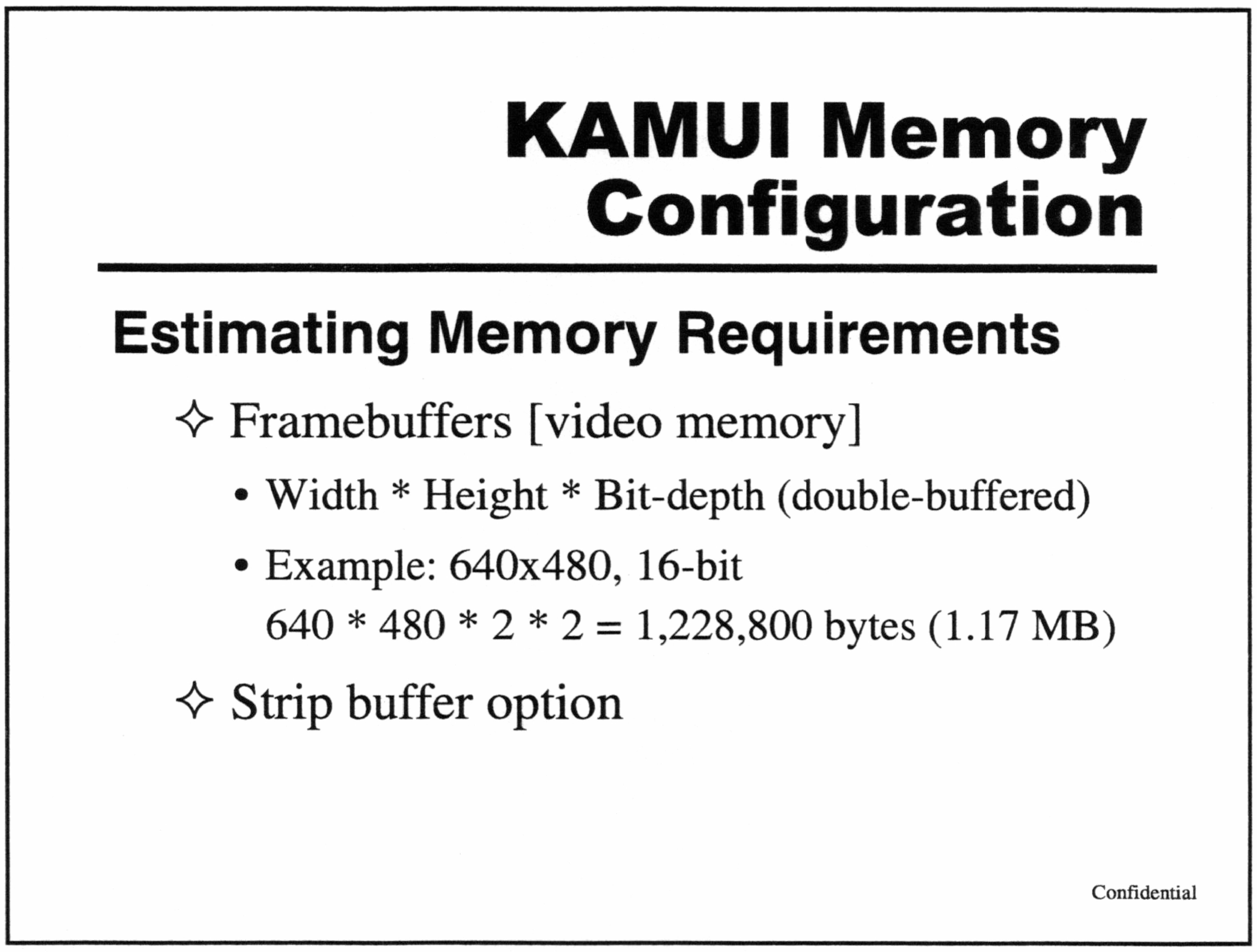
Confidential



KAMUI Memory Configuration

- Objects = ISP/TSP parameter words and vertices that define an internal triangle strip
- Global Parameter = Render state information
- Control Parameter = User clip & object settings
- ISP = Image Synthesis Processor
- TSP = Texture and Shading Processor
- Texture Parameter = Texture surface description

Confidential



KAMUI Memory Configuration

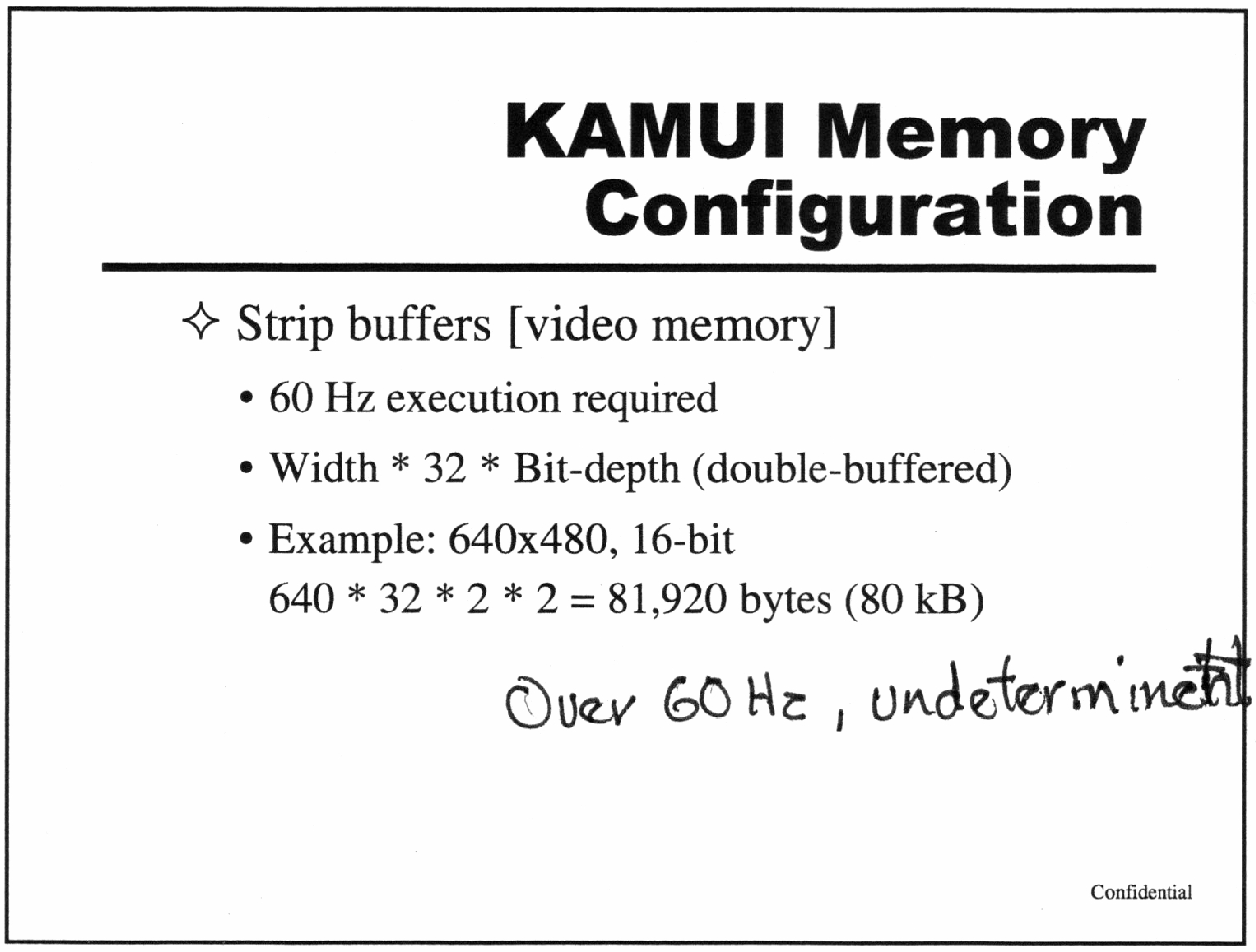
Estimating Memory Requirements

✧ Framebuffers [video memory]

- Width * Height * Bit-depth (double-buffered)
- Example: 640x480, 16-bit
  640 * 480 * 2 * 2 = 1,228,800 bytes (1.17 MB)

✧ Strip buffer option

Confidential



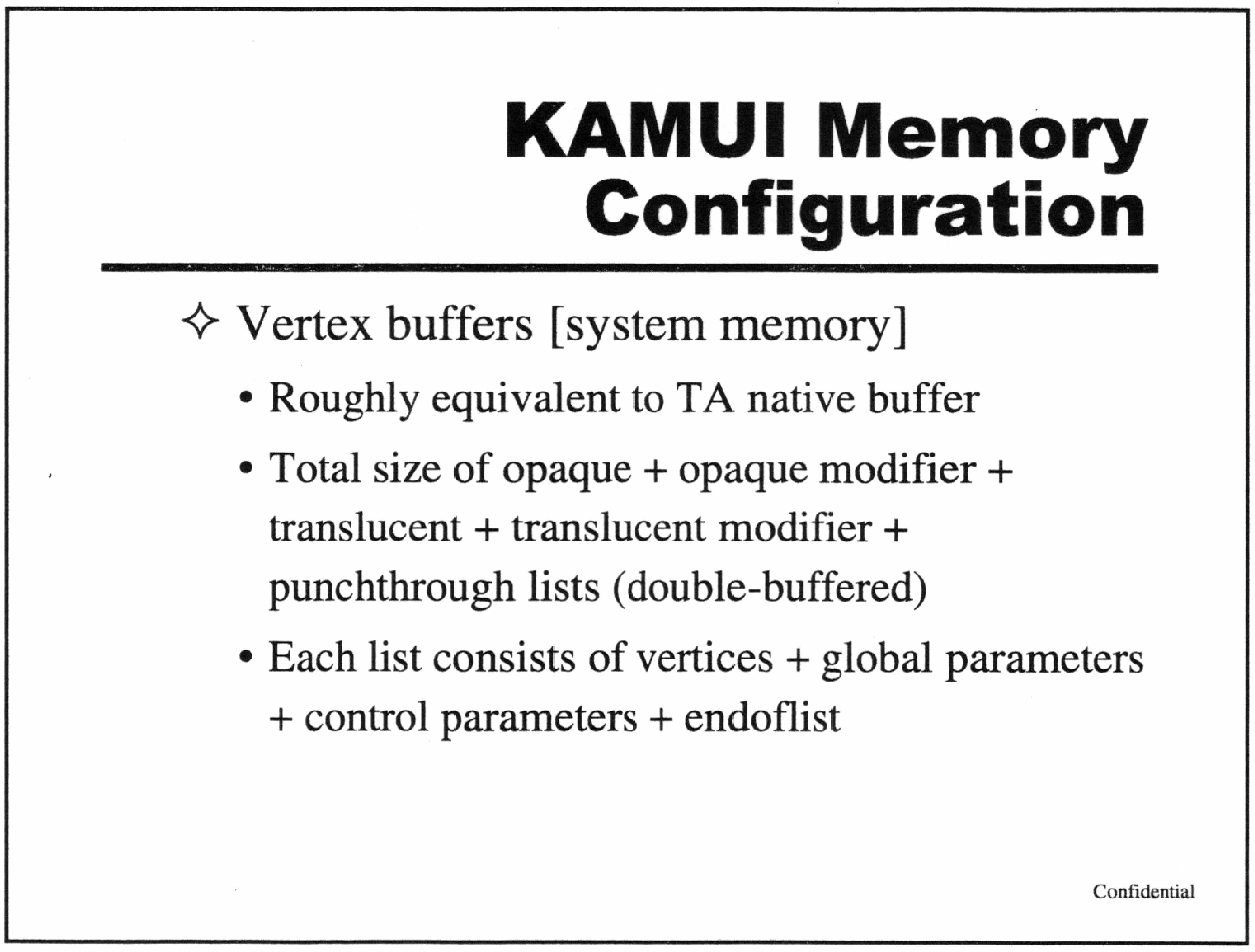
KAMUI Memory Configuration

✧ Strip buffers [video memory]

- 60 Hz execution required
- Width * 32 * Bit-depth (double-buffered)
- Example: 640x480, 16-bit
  640 * 32 * 2 * 2 = 81,920 bytes (80 kB)

Over 60 Hz, undeterminent results

Confidential

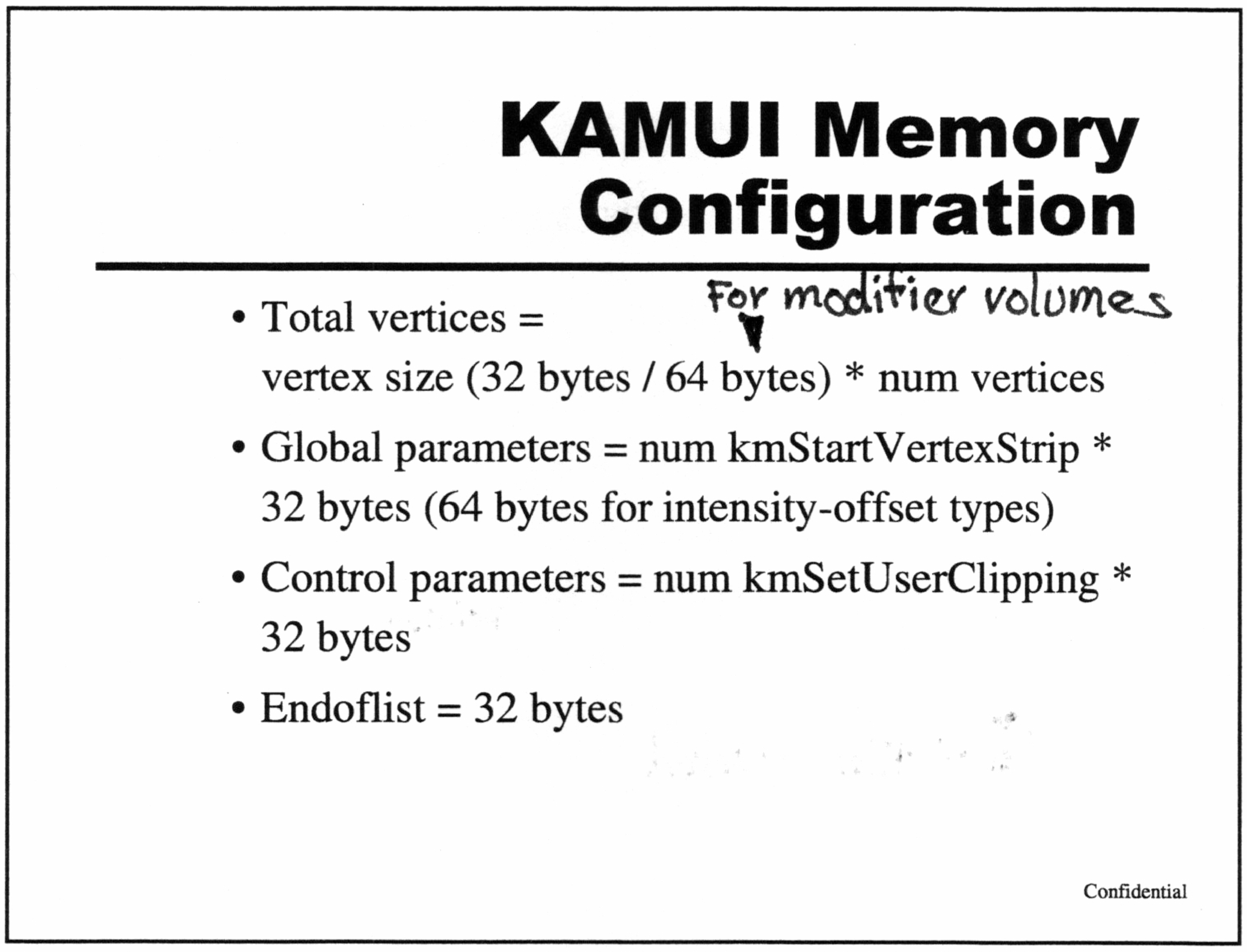


KAMUI Memory Configuration

✧ Vertex buffers [system memory]

- Roughly equivalent to TA native buffer
- Total size of opaque + opaque modifier + translucent + translucent modifier + punchthrough lists (double-buffered)
- Each list consists of vertices + global parameters + control parameters + endoflist

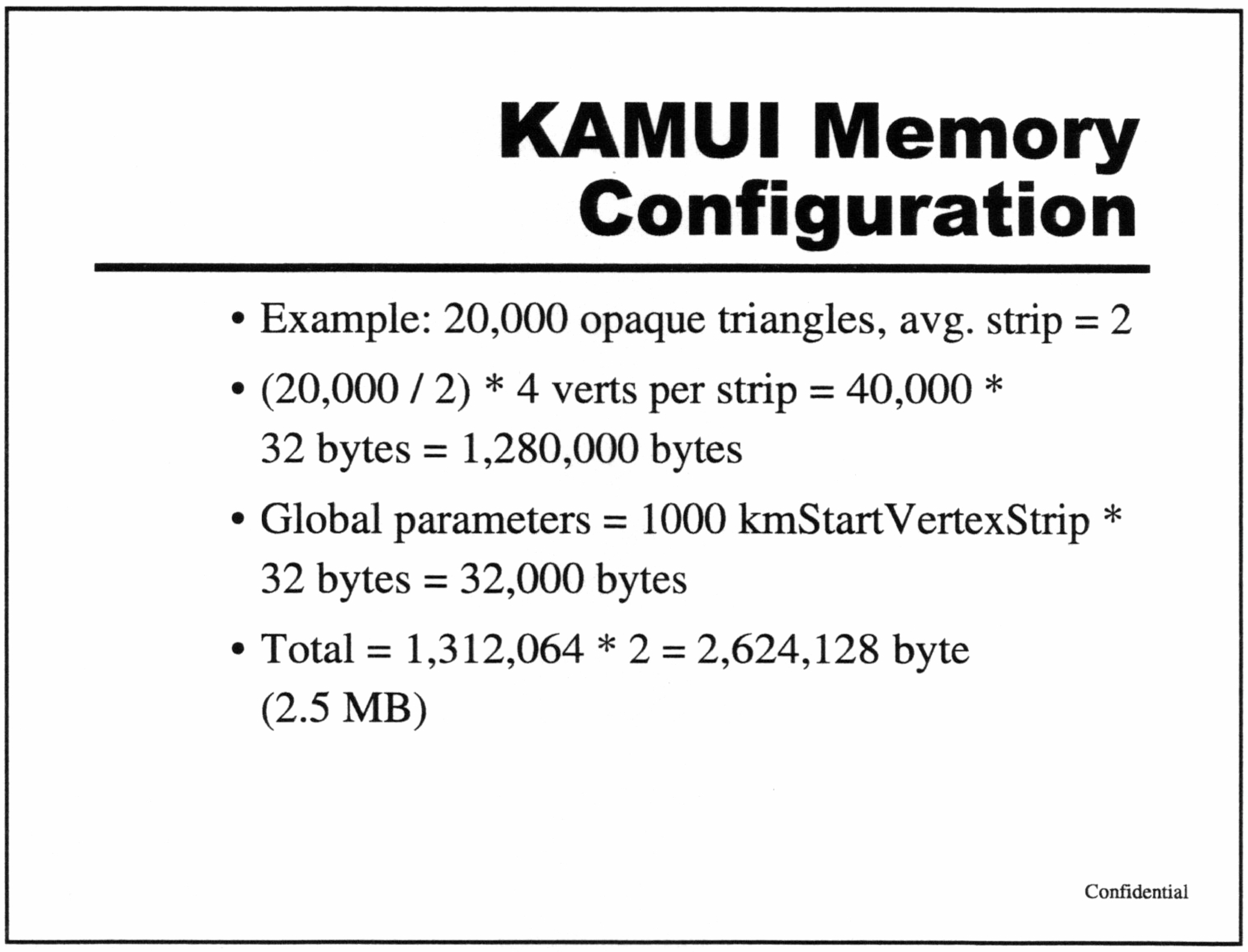
Confidential



KAMUI Memory Configuration

- Total vertices = For modifier volumes
  vertex size (32 bytes / 64 bytes) * num vertices
- Global parameters = num kmStartVertexStrip * 32 bytes (64 bytes for intensity-offset types)
- Control parameters = num kmSetUserClipping * 32 bytes
- Endoflist = 32 bytes

Confidential



KAMUI Memory Configuration

- Example: 20,000 opaque triangles, avg. strip = 2
- (20,000 / 2) * 4 verts per strip = 40,000 * 32 bytes = 1,280,000 bytes
- Global parameters = 1000 kmStartVertexStrip * 32 bytes = 32,000 bytes
- Total = 1,312,064 * 2 = 2,624,128 byte (2.5 MB)

Confidential

KAMUI Memory Configuration

✧ Native buffer [video memory]

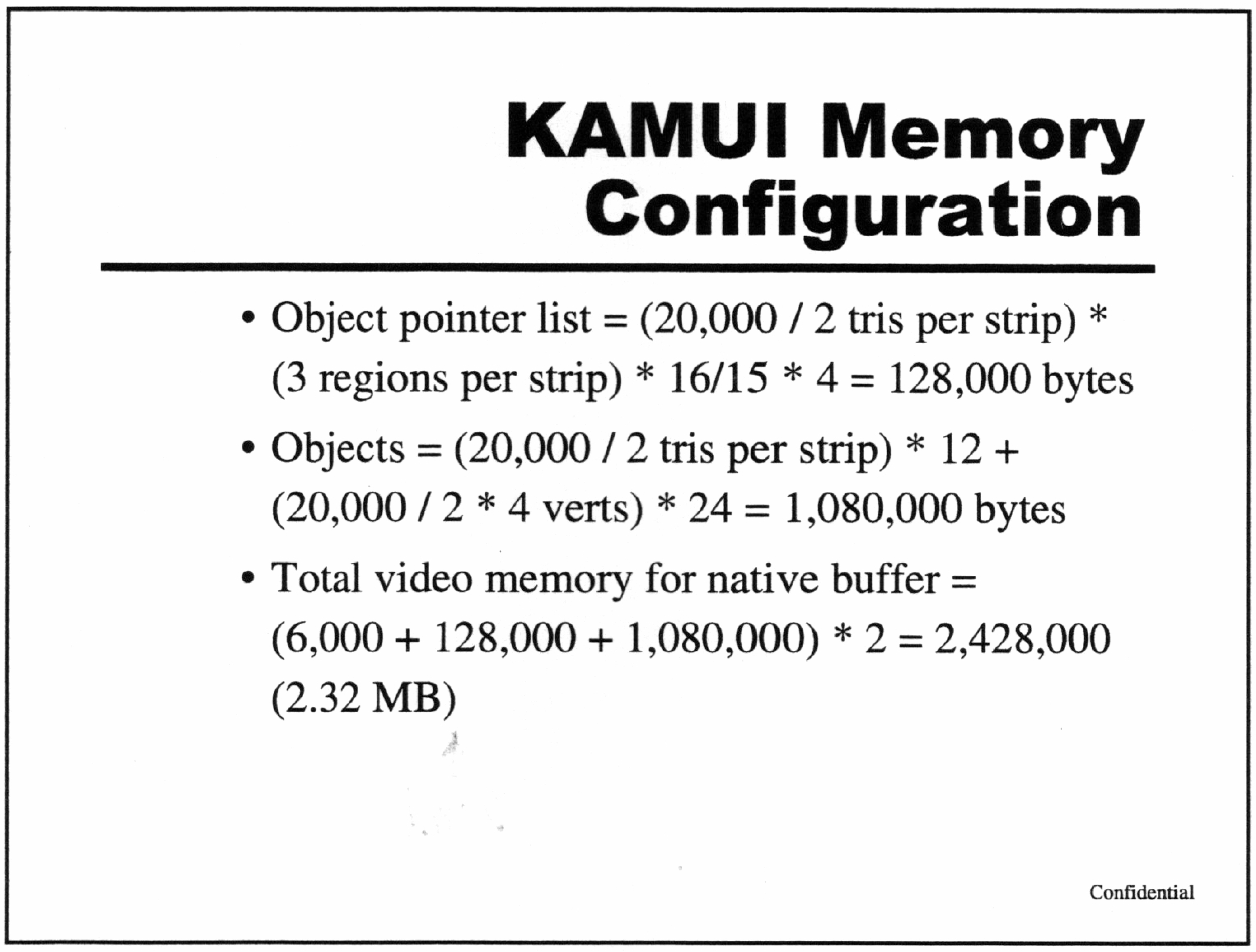
- Total size equivalent to vertex buffers plus overhead for region arrays and object lists
- Region array size = (framebuffer width / 32) * (framebuffer height / 32) * 5 * 4 bytes

  Pointers
- Object pointer list = (total internal strips) * (avg. regions per strip) * 16/15 * 4 bytes

  # of tiles covered

KAMUI Memory Configuration

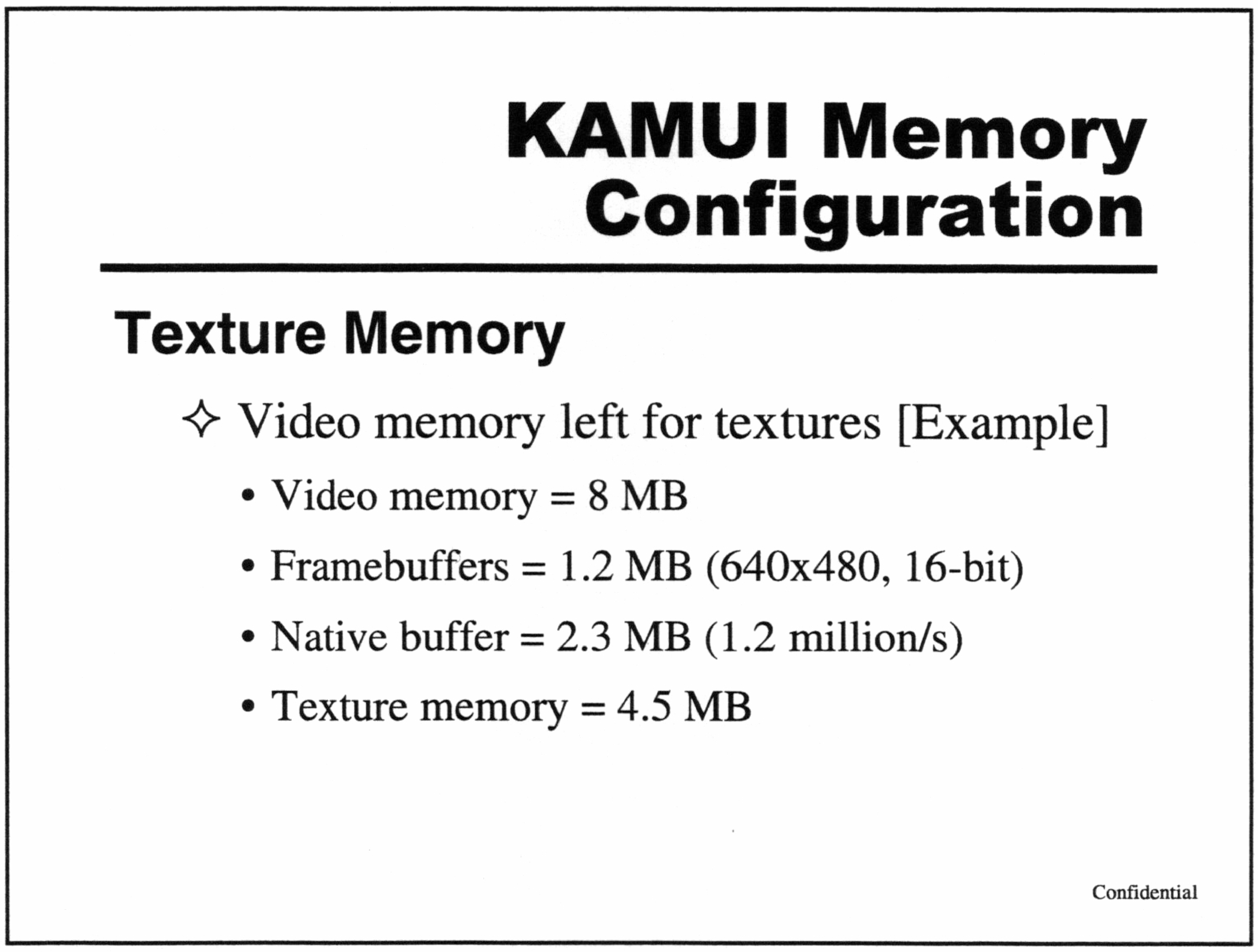
- Objects = (total internal strips) * (TSP data) + (total vertices) * (ISP vertex data)
- ISP = vertex size - padding, TSP = 12 - 24 bytes
- Example: 20,000 opaque triangles, internal strip length = 2, avg. regions per strip = 3
- Region array size = (640 / 32) * (480 / 32) * 20 bytes = 6,000 bytes



KAMUI Memory Configuration

- Object pointer list = (20,000 / 2 tris per strip) * (3 regions per strip) * 16/15 * 4 = 128,000 bytes
- Objects = (20,000 / 2 tris per strip) * 12 + (20,000 / 2 * 4 verts) * 24 = 1,080,000 bytes
- Total video memory for native buffer = (6,000 + 128,000 + 1,080,000) * 2 = 2,428,000 (2.32 MB)

Confidential



KAMUI Memory Configuration

Texture Memory

✧ Video memory left for textures [Example]

- Video memory = 8 MB
- Framebuffers = 1.2 MB (640x480, 16-bit)
- Native buffer = 2.3 MB (1.2 million/s)
- Texture memory = 4.5 MB

Confidential

KAMUI Texture Management

Loading Textures

✧ kmLoadTexture() requirements

- DMA transfer from system to video memory
- No mechanism for loading textures directly from GD-ROM to video memory
- 32-byte alignment (including 16 byte PVRT chunk header)

↑ Dumb

Confidential

KAMUI Texture Management

Partial Texture Loads

✧ Simulate GD-ROM to video mem. load

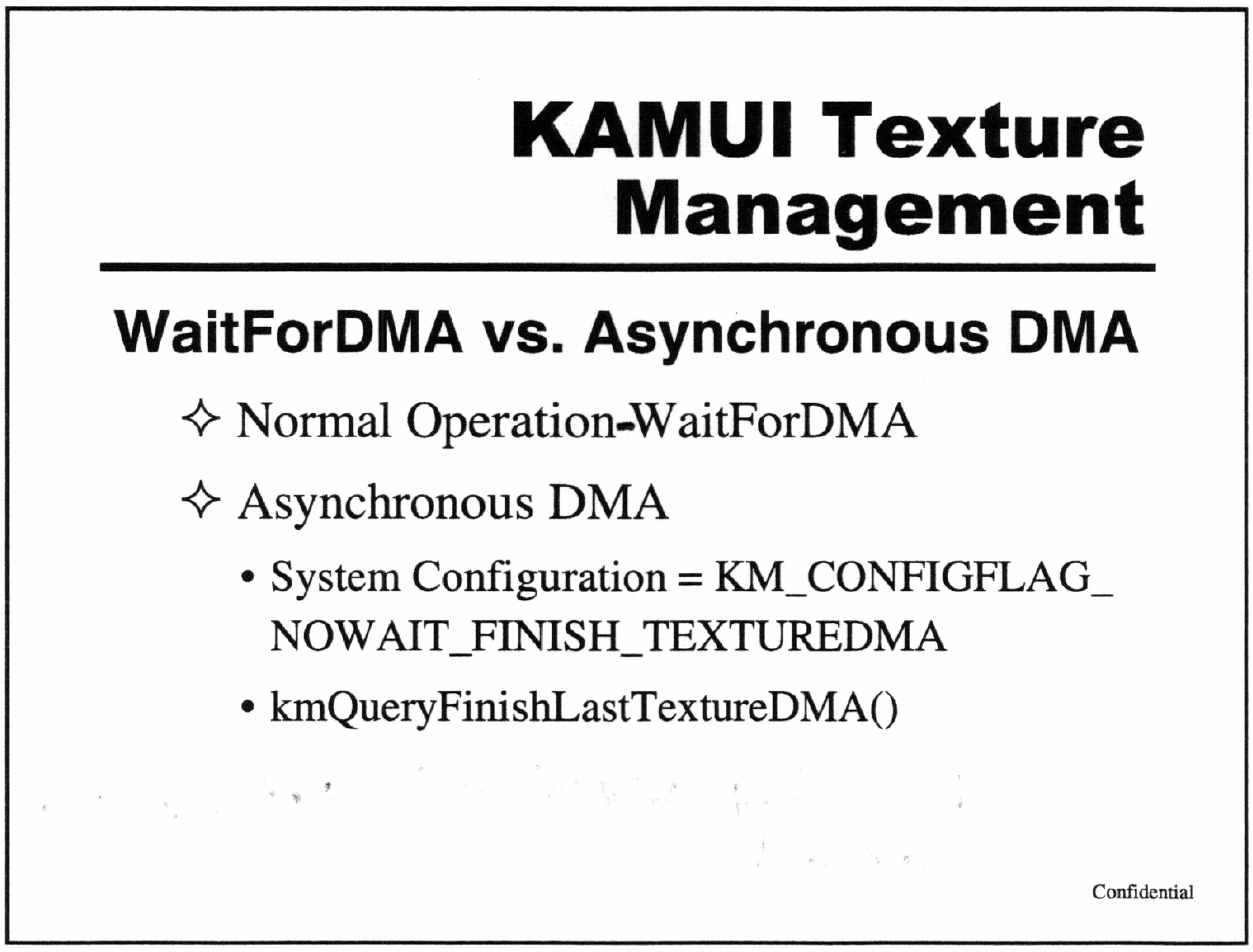
✧ kmLoadTextureBlock()

- Fixed block size of 32 bytes

✧ kmLoadTexturePart()

- Variable size sections (multiple of 32 bytes)

Confidential

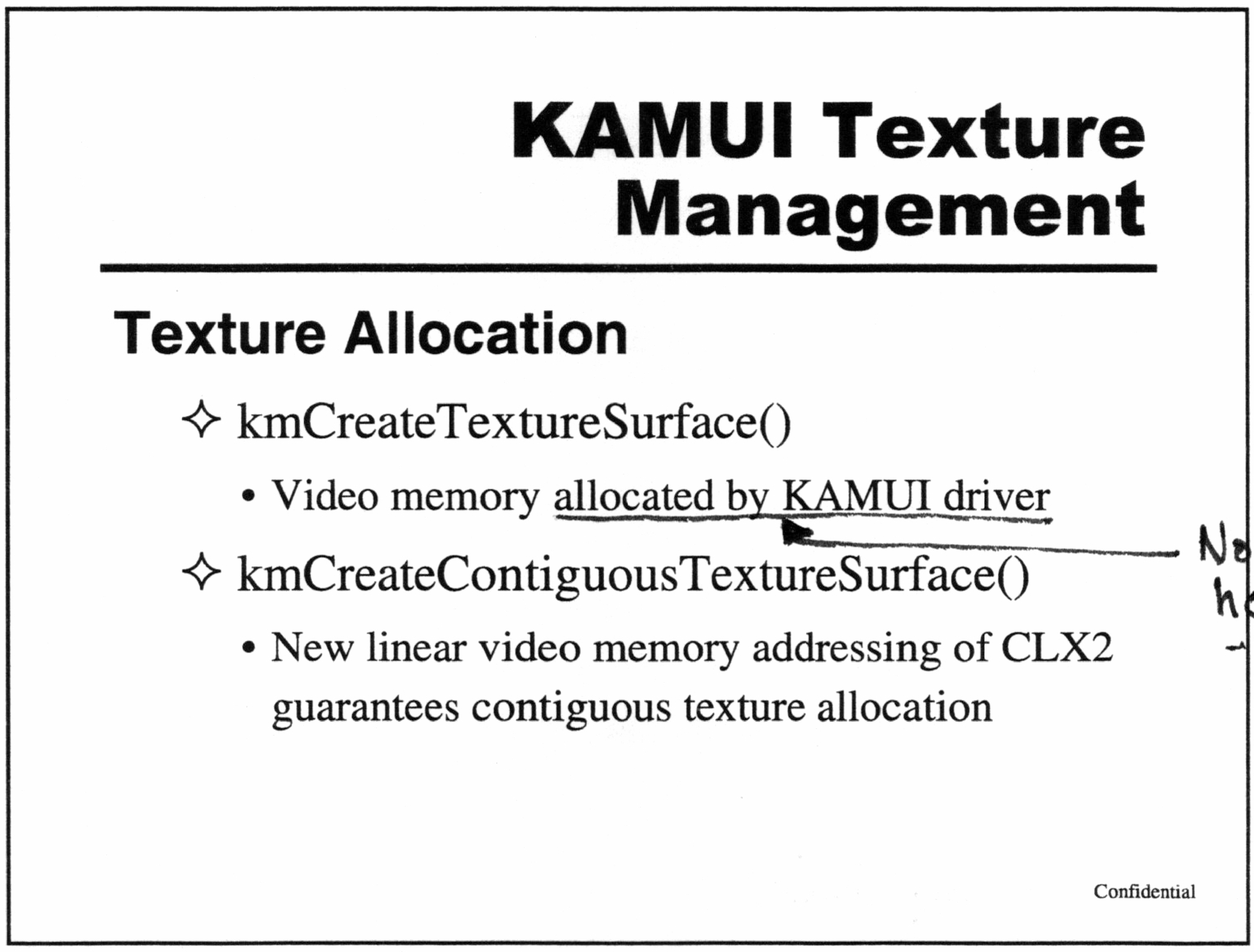


KAMUI Texture Management

WaitForDMA vs. Asynchronous DMA

- ✧ Normal Operation-WaitForDMA
- ✧ Asynchronous DMA
  - System Configuration = KM_CONFIGFLAG_NOWAIT_FINISH_TEXTUREDMA
  - kmQueryFinishLastTextureDMA()

Confidential



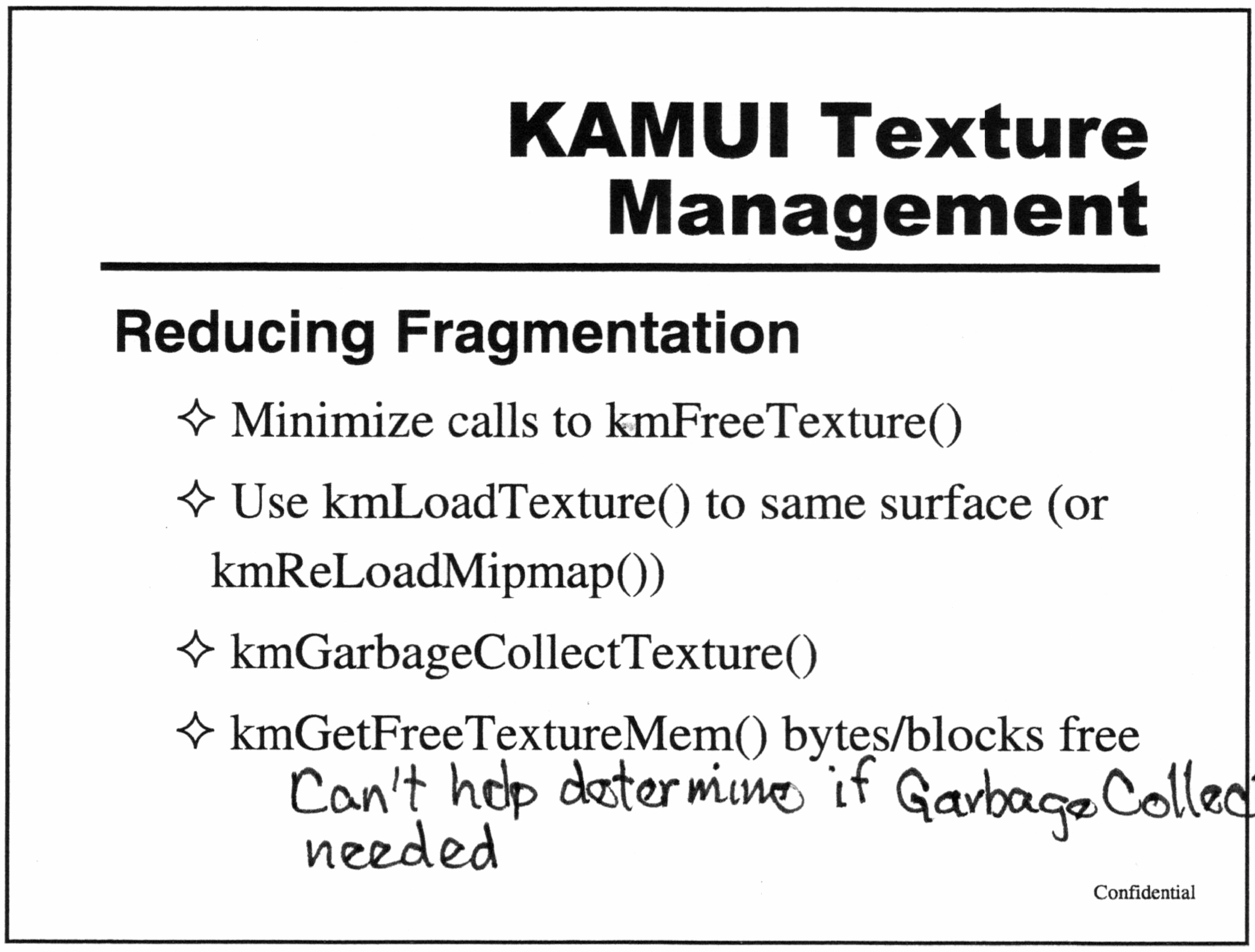
KAMUI Texture Management

Texture Allocation

- ✧ kmCreateTextureSurface()
  - Video memory allocated by KAMUI driver
- ✧ kmCreateContiguousTextureSurface()
  - New linear video memory addressing of CLX2 guarantees contiguous texture allocation

Confidential

No user control of how memory is carved up.



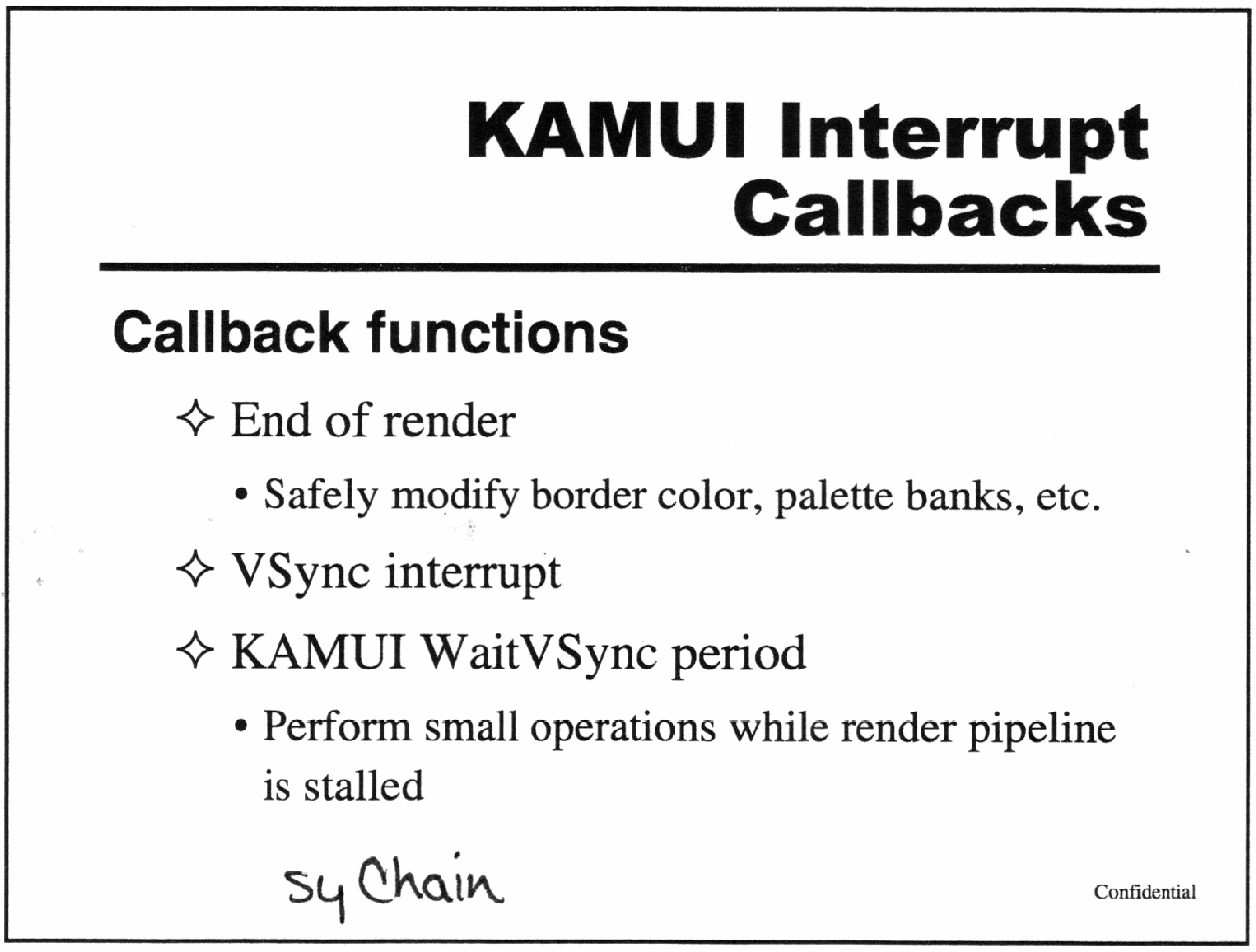
KAMUI Texture Management

Reducing Fragmentation

- ✧ Minimize calls to kmFreeTexture()
- ✧ Use kmLoadTexture() to same surface (or kmReLoadMipmap())
- ✧ kmGarbageCollectTexture()
- ✧ kmGetFreeTextureMem() bytes/blocks free

Can't help determine if GarbageCollection is needed

Confidential



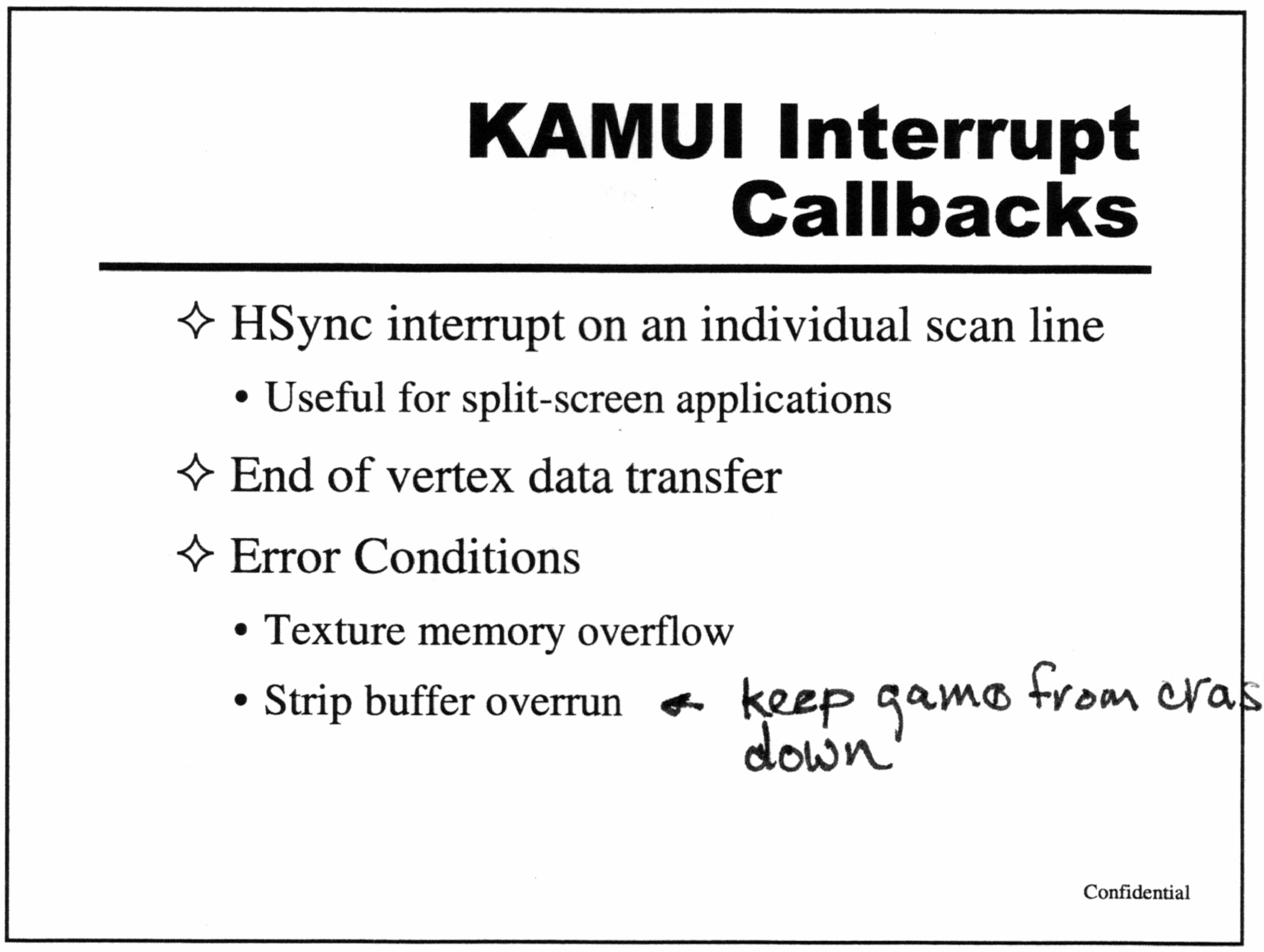
KAMUI Interrupt Callbacks

Callback functions

- ✧ End of render
  - Safely modify border color, palette banks, etc.
- ✧ VSync interrupt
- ✧ KAMUI WaitVSync period
  - Perform small operations while render pipeline is stalled

Sy Chain

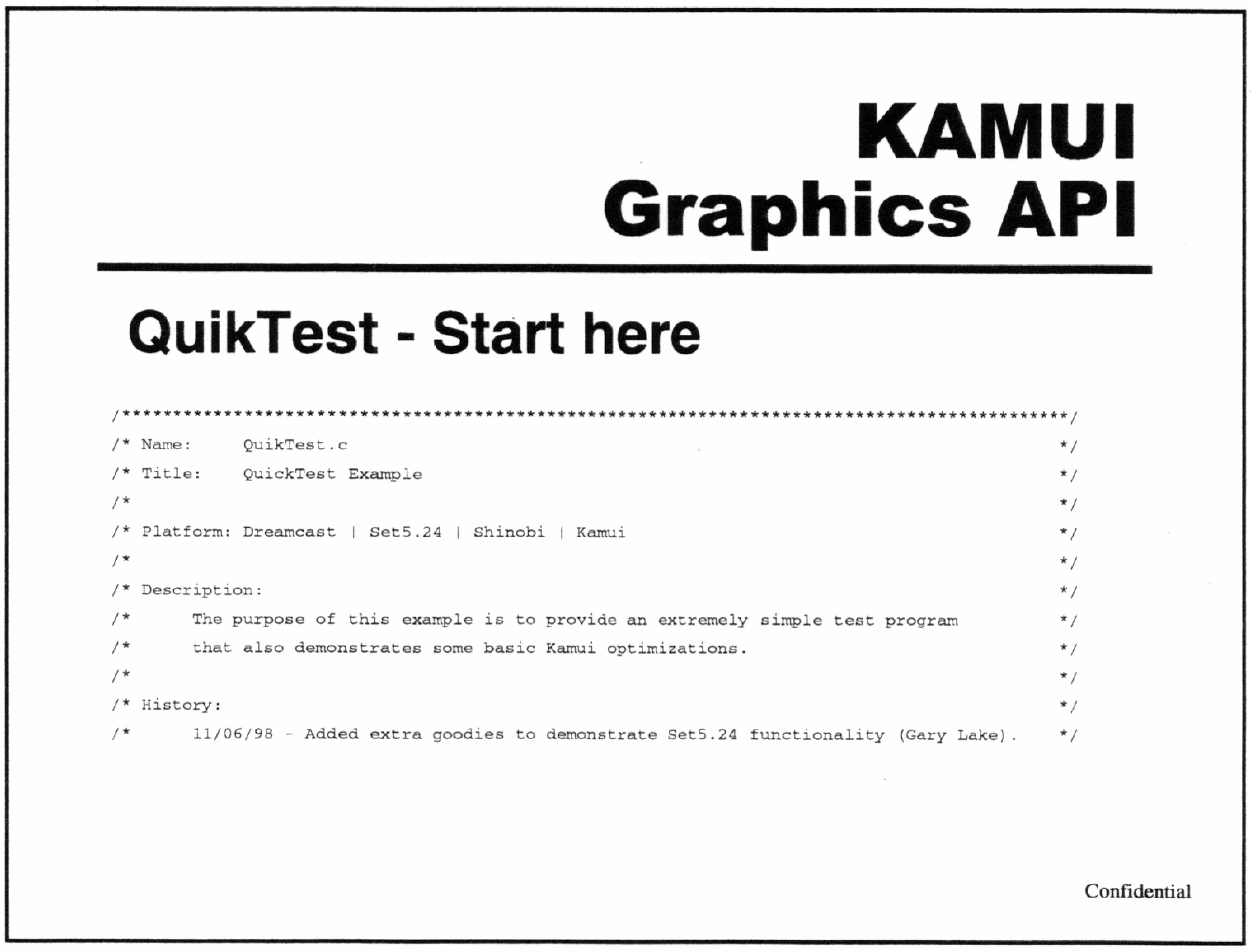
Confidential



KAMUI Interrupt Callbacks

- ✧ HSync interrupt on an individual scan line
  - Useful for split-screen applications
- ✧ End of vertex data transfer
- ✧ Error Conditions
  - Texture memory overflow
  - Strip buffer overrun ← keep game from crashing if slowed down

Confidential

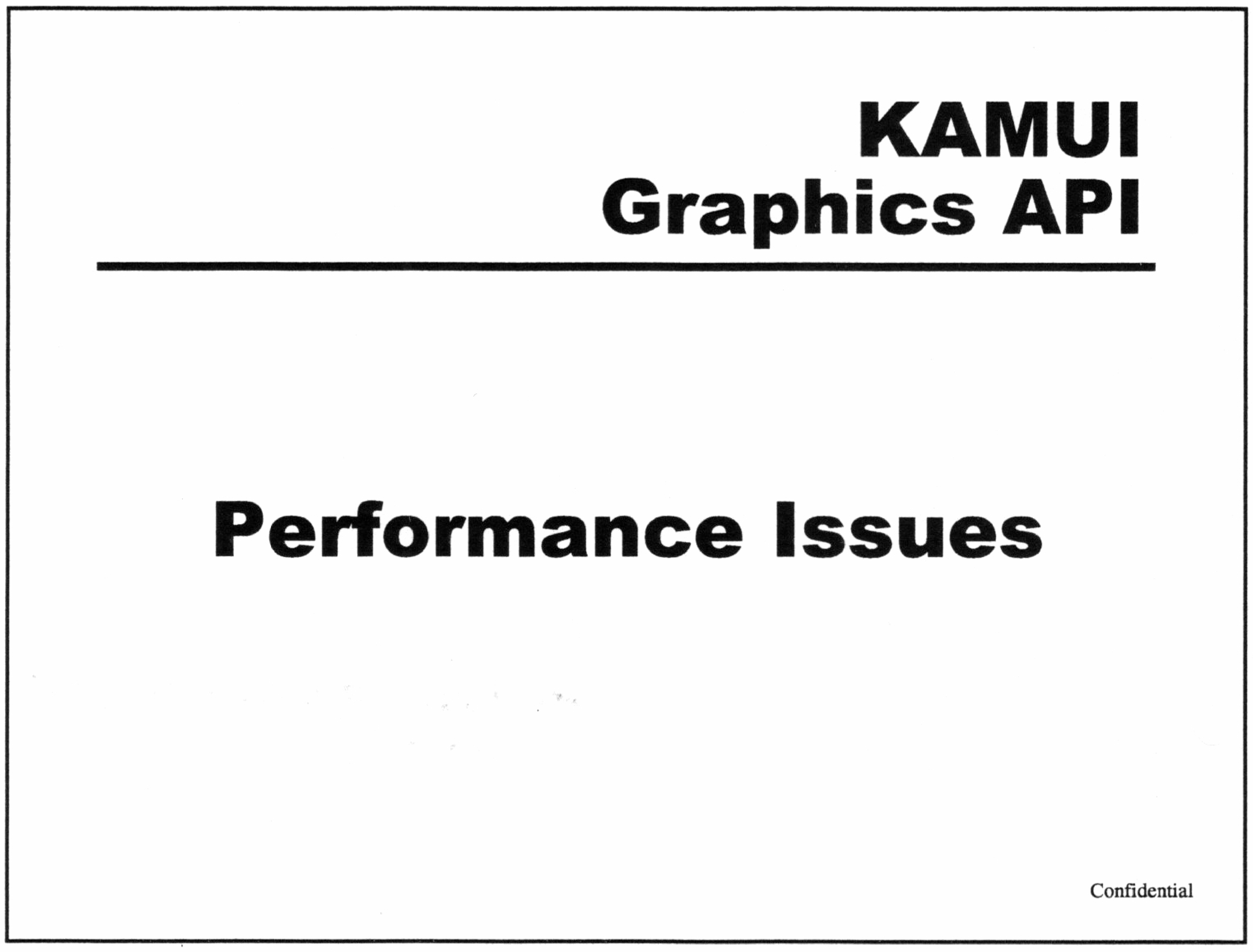


KAMUI Graphics API

QuikTest - Start here

```
/**********************************************************************************/
/* Name:     QuikTest.c                                                          */
/* Title:    QuickTest Example                                                   */
/*                                                                               */
/* Platform: Dreamcast | Set5.24 | Shinobi | Kamui                               */
/*                                                                               */
/* Description:                                                                  */
/*      The purpose of this example is to provide an extremely simple test program */
/*      that also demonstrates some basic Kamui optimizations.                   */
/*                                                                               */
/* History:                                                                      */
/*      11/06/98 - Added extra goodies to demonstrate Set5.24 functionality (Gary Lake).  */
```

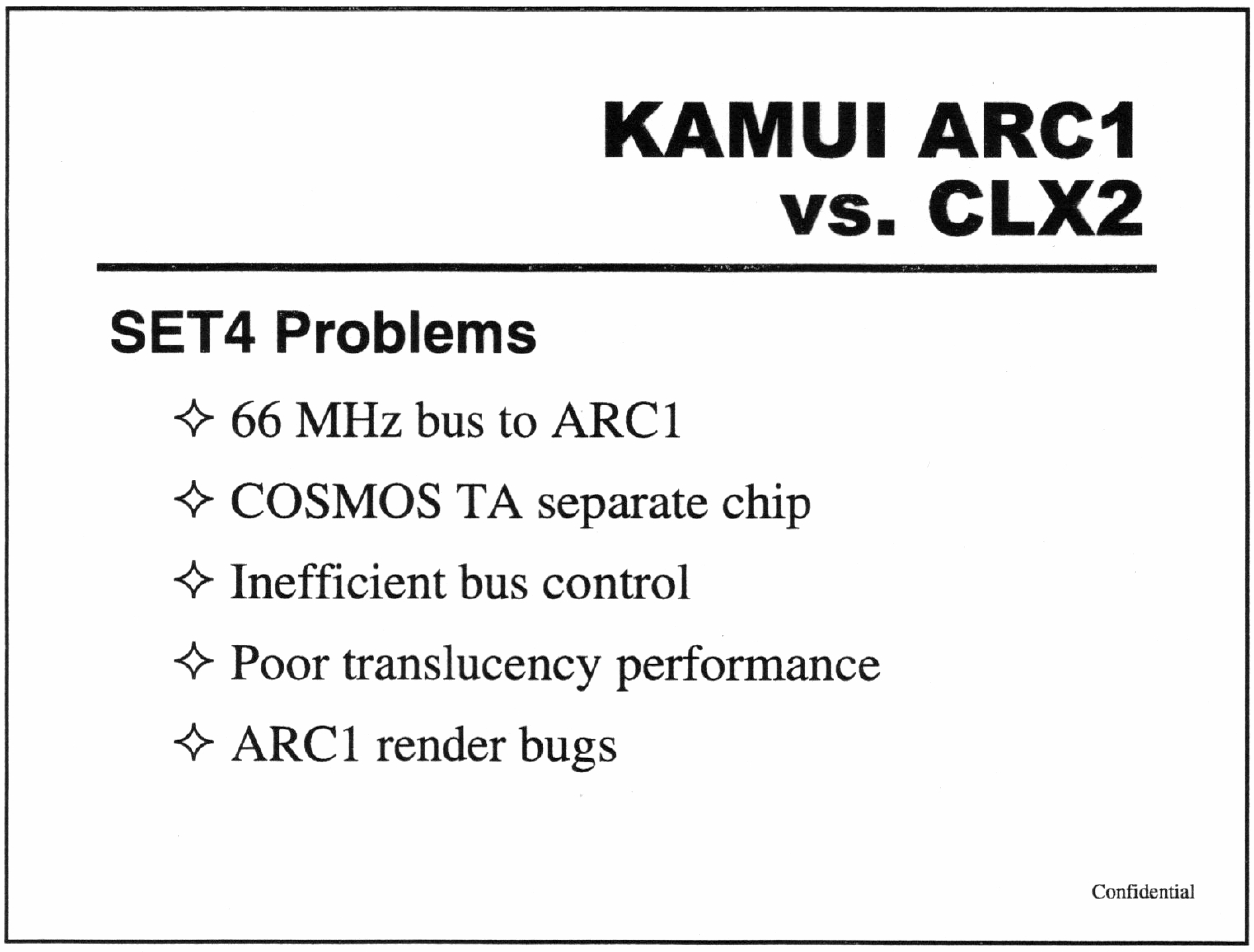
Confidential



KAMUI
Graphics API

Performance Issues

Confidential



KAMUI ARC1
vs. CLX2

SET4 Problems

✧ 66 MHz bus to ARC1

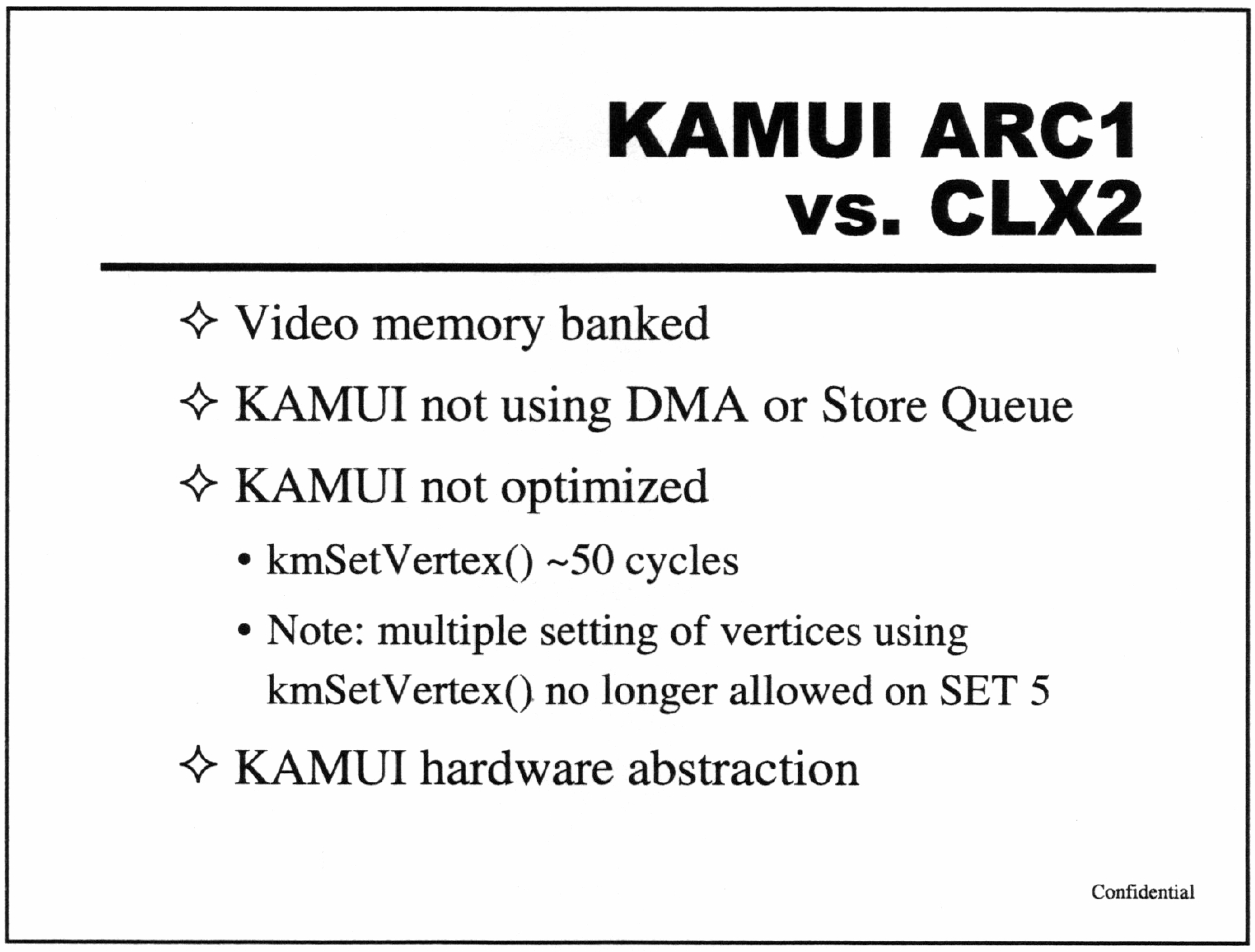
✧ COSMOS TA separate chip

✧ Inefficient bus control

✧ Poor translucency performance

✧ ARC1 render bugs

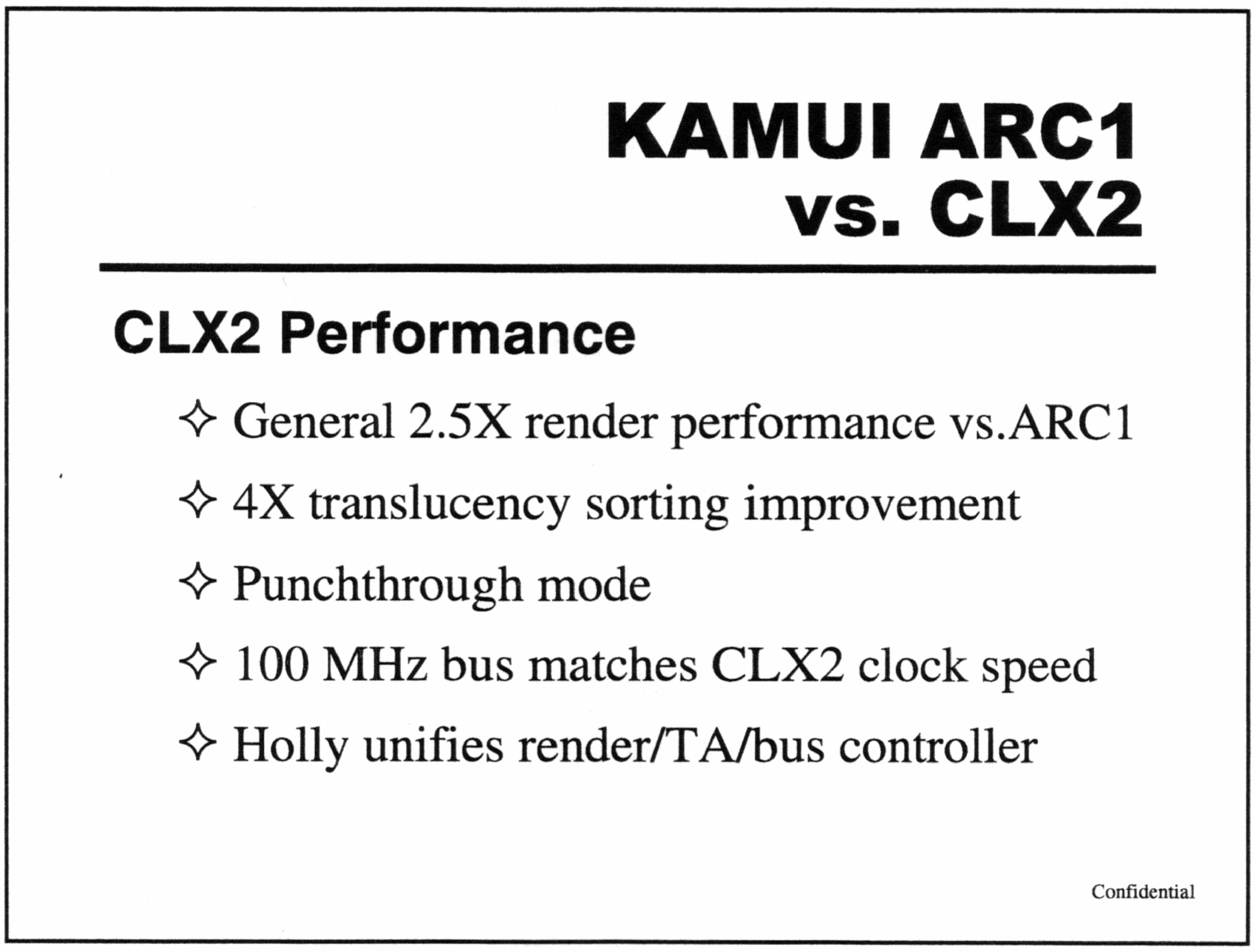
Confidential



KAMUI ARC1 vs. CLX2

- ✧ Video memory banked
- ✧ KAMUI not using DMA or Store Queue
- ✧ KAMUI not optimized
  - kmSetVertex() ~50 cycles
  - Note: multiple setting of vertices using kmSetVertex() no longer allowed on SET 5
- ✧ KAMUI hardware abstraction

Confidential

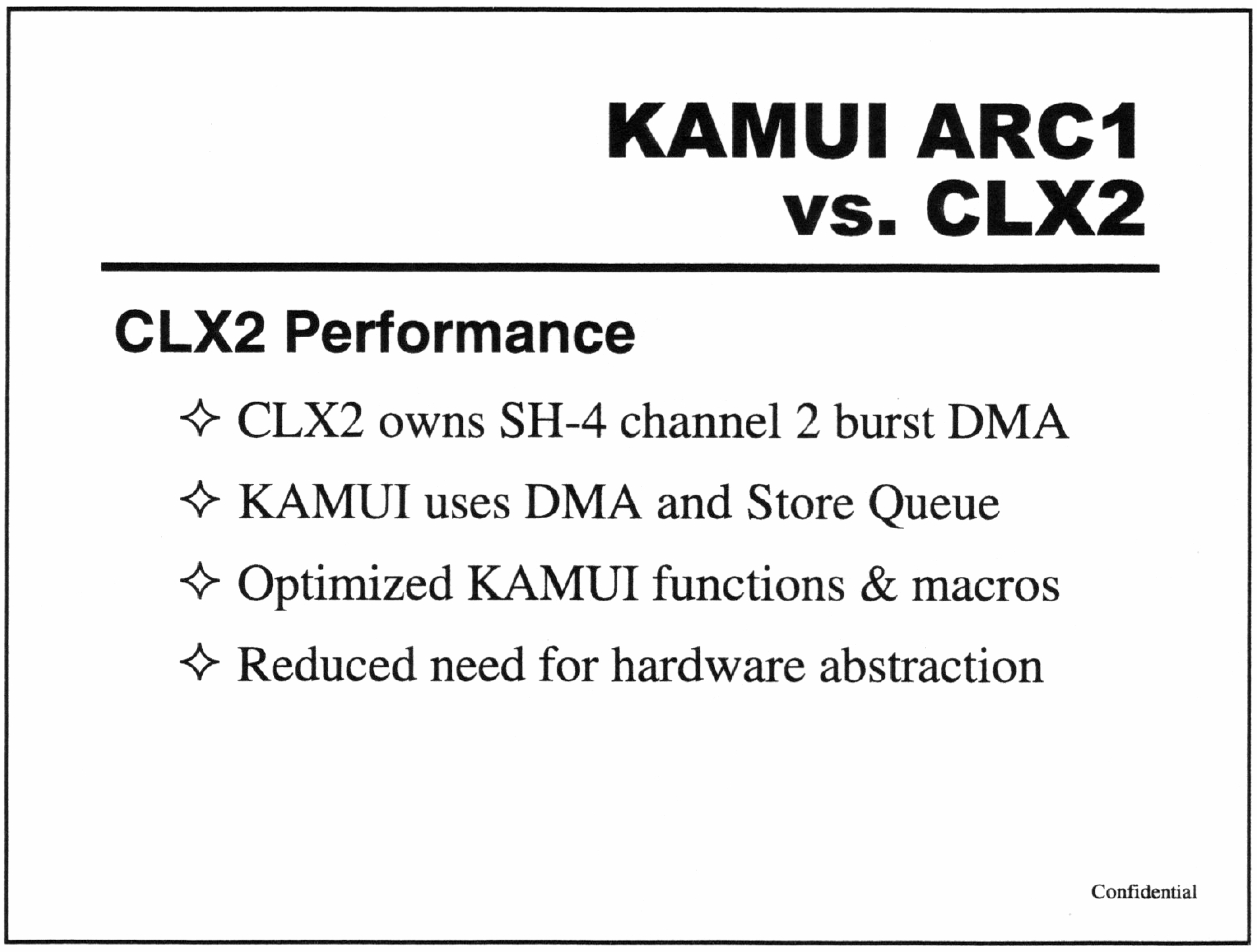


KAMUI ARC1 vs. CLX2

CLX2 Performance

- ✧ General 2.5X render performance vs.ARC1
- ✧ 4X translucency sorting improvement
- ✧ Punchthrough mode
- ✧ 100 MHz bus matches CLX2 clock speed
- ✧ Holly unifies render/TA/bus controller

Confidential

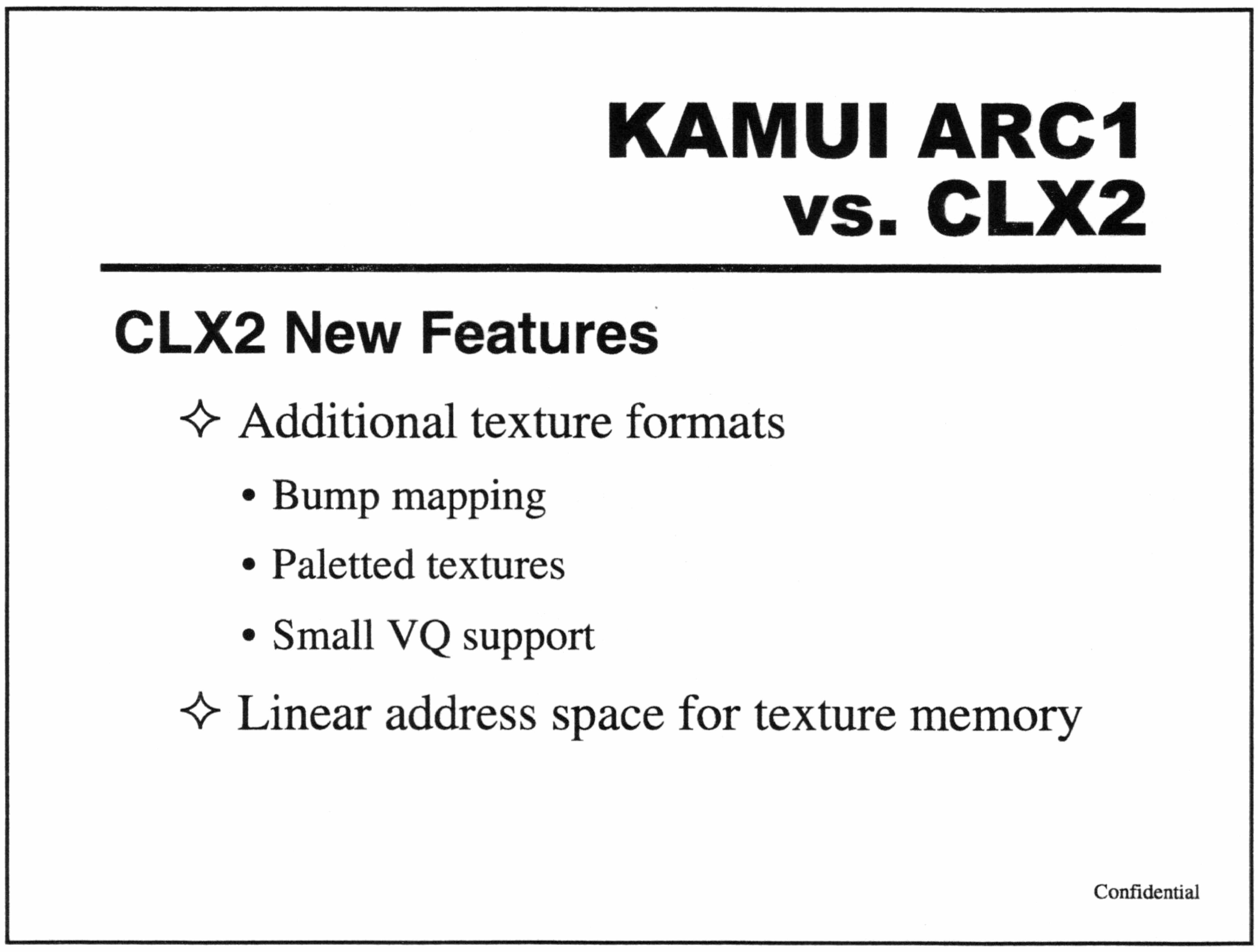


KAMUI ARC1 vs. CLX2

CLX2 Performance

- ✧ CLX2 owns SH-4 channel 2 burst DMA
- ✧ KAMUI uses DMA and Store Queue
- ✧ Optimized KAMUI functions & macros
- ✧ Reduced need for hardware abstraction

Confidential

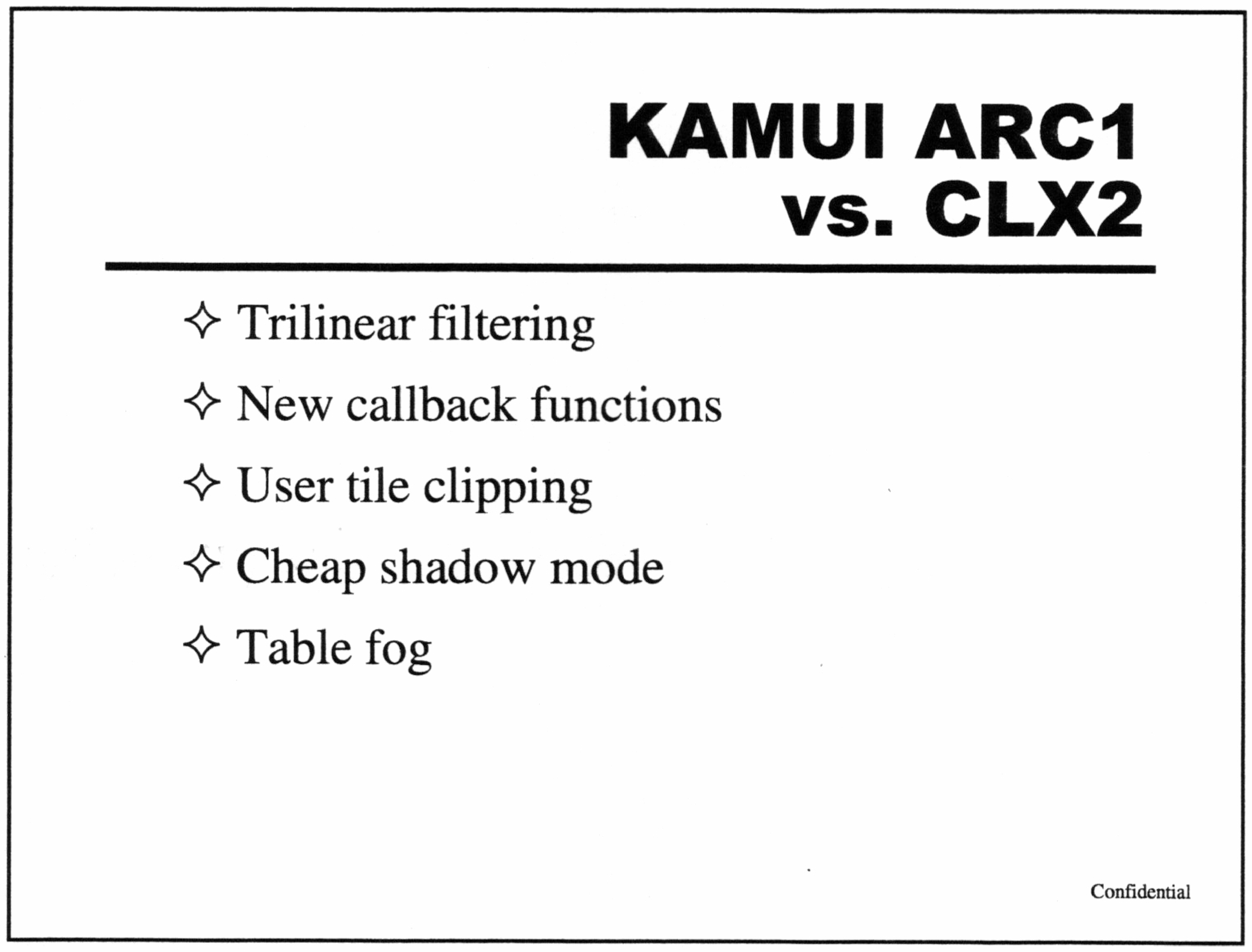


KAMUI ARC1 vs. CLX2

CLX2 New Features

- ✧ Additional texture formats
  - Bump mapping
  - Paletted textures
  - Small VQ support
- ✧ Linear address space for texture memory

Confidential



KAMUI ARC1 vs. CLX2

✧ Trilinear filtering

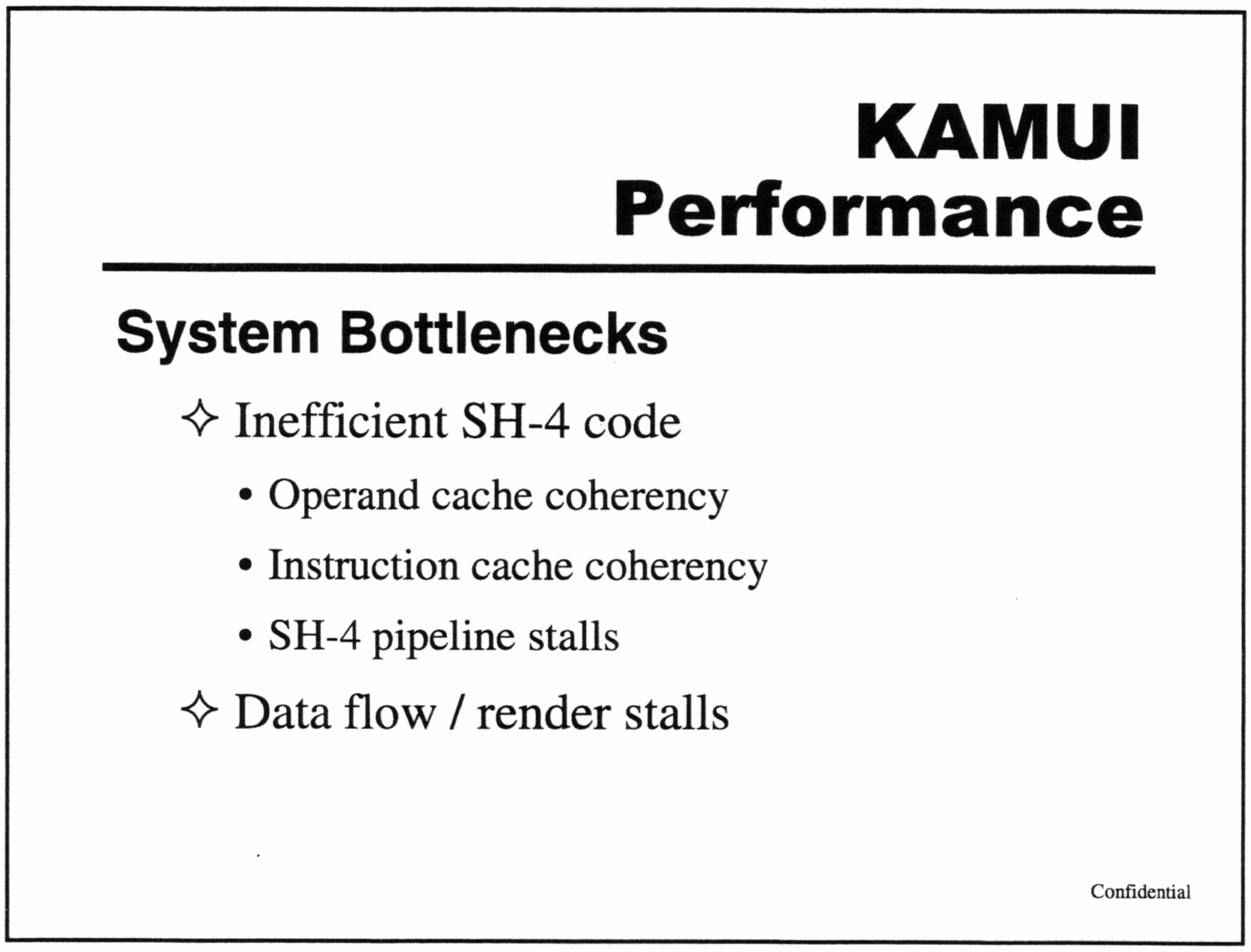
✧ New callback functions

✧ User tile clipping

✧ Cheap shadow mode

✧ Table fog

Confidential



KAMUI Performance

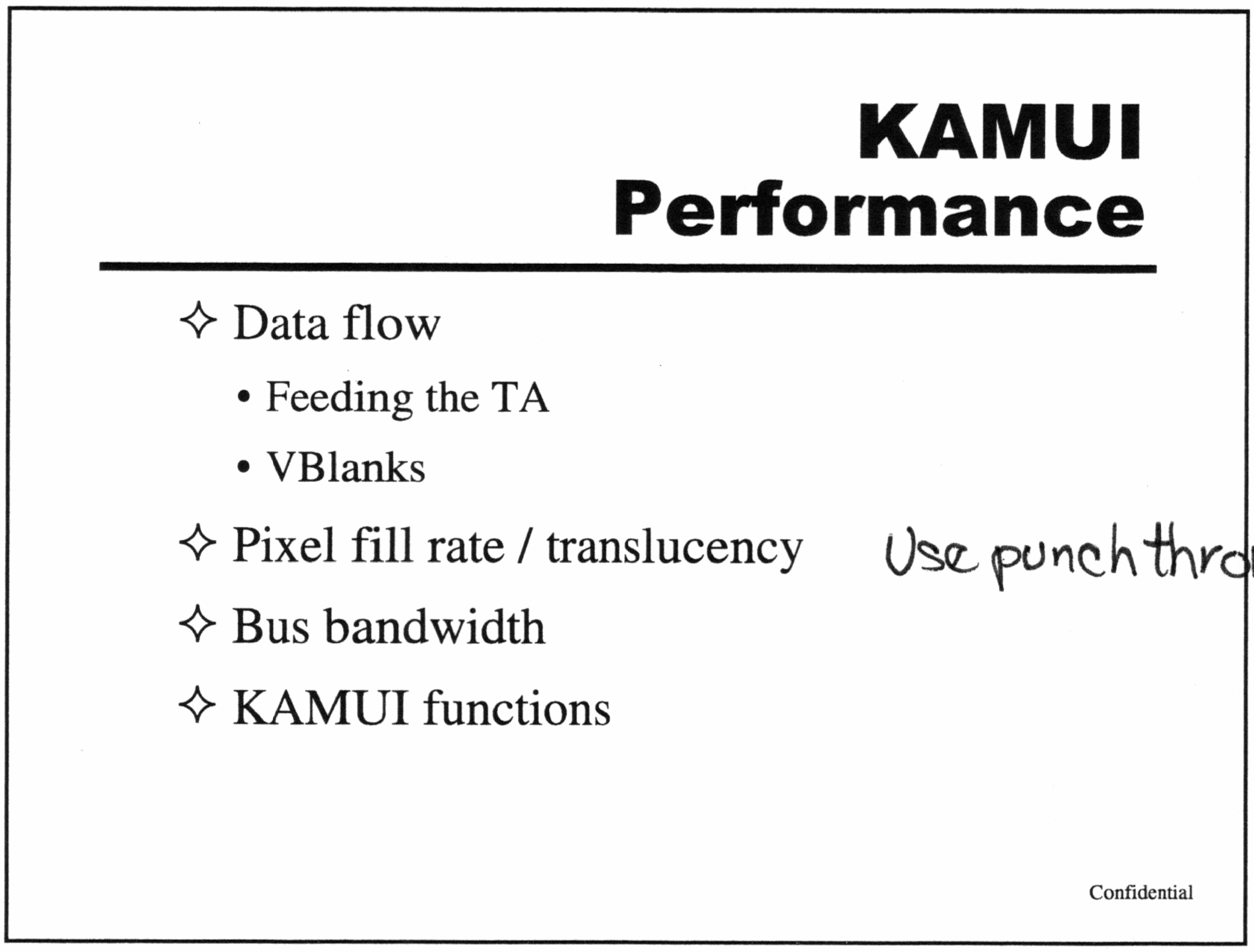
System Bottlenecks

✧ Inefficient SH-4 code

- Operand cache coherency
- Instruction cache coherency
- SH-4 pipeline stalls

✧ Data flow / render stalls

Confidential

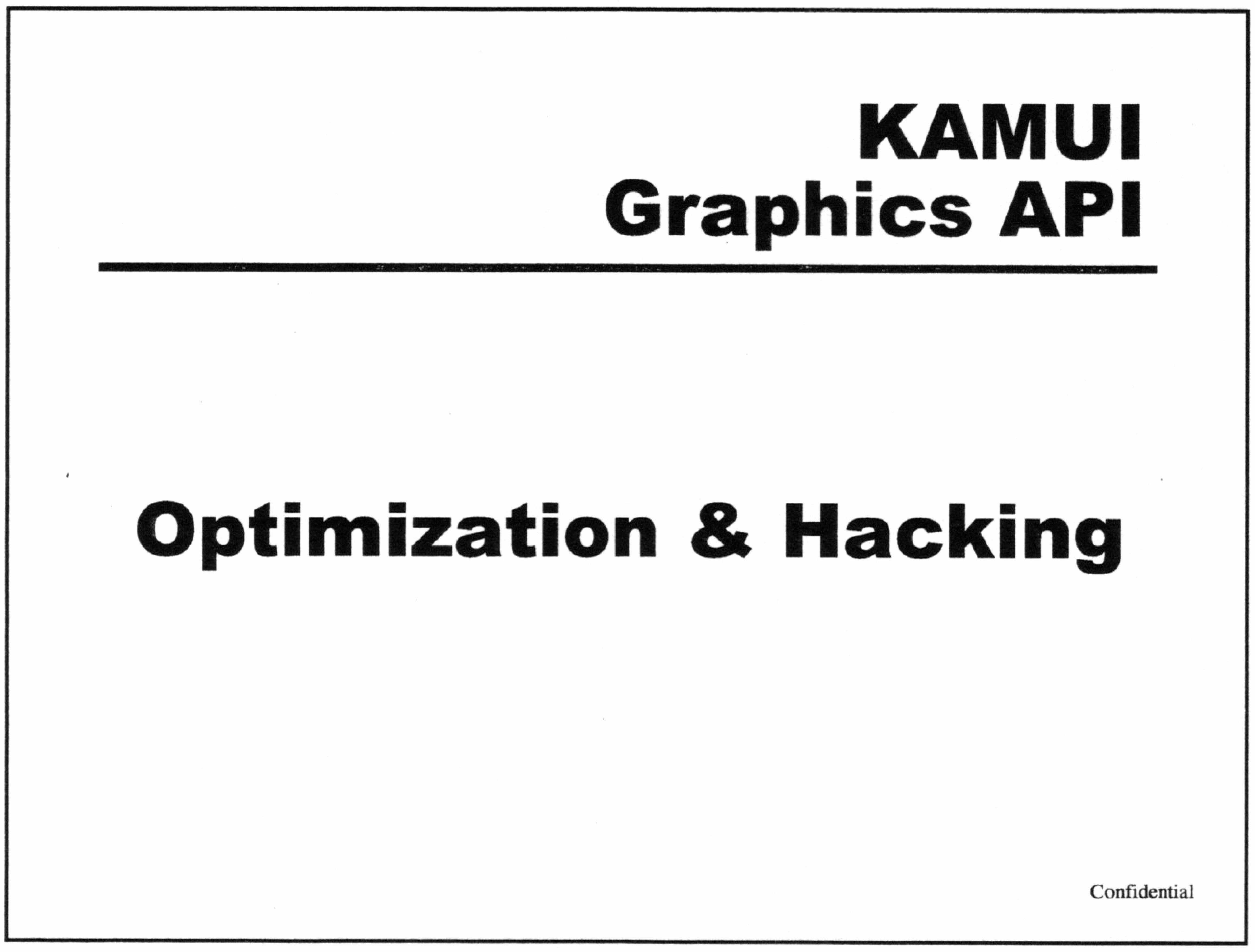


KAMUI Performance

- ✧ Data flow
  - Feeding the TA
  - VBlanks
- ✧ Pixel fill rate / translucency
- ✧ Bus bandwidth
- ✧ KAMUI functions

Confidential

Use punch through mode



KAMUI Graphics API

Optimization & Hacking

Confidential

KAMUI What to Optimize?

SH-4 Matrix Transformations

- ✧ 4x4 matrix multiplier [FTRV]
  - 12 cycle (goal)
- ✧ Inner product [FIPR]
- ✧ Sin-cos approximation [FSCA]
- ✧ 1/square-root approximation [FSRRA]

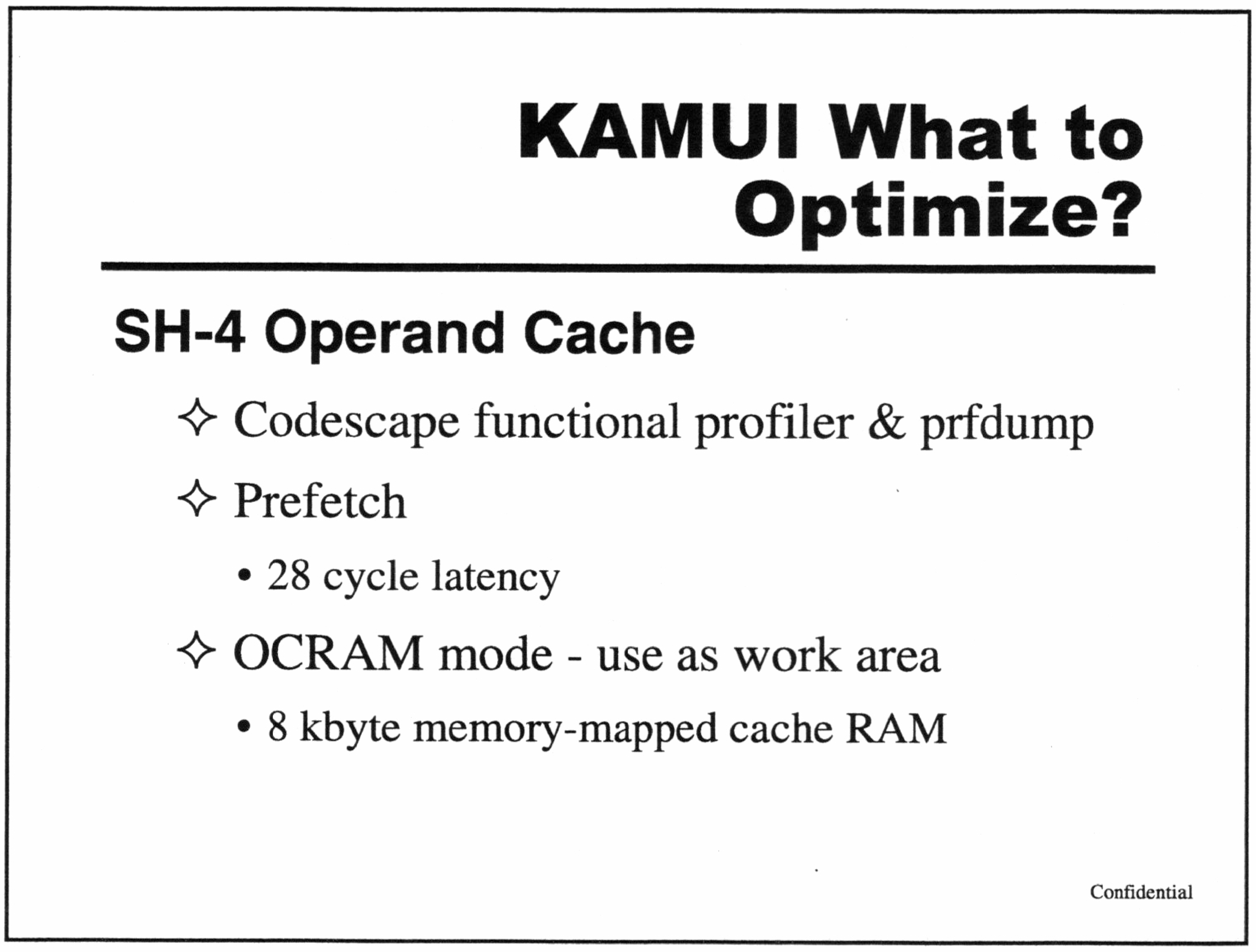
KAMUI What to Optimize?

SH-4 Pipeline

- ✧ Codescape simulator

SH-4 Instruction Cache

- ✧ Keep loop size under 8 kbytes
- ✧ Avoid jumping around in memory (i.e. excessive function calls)

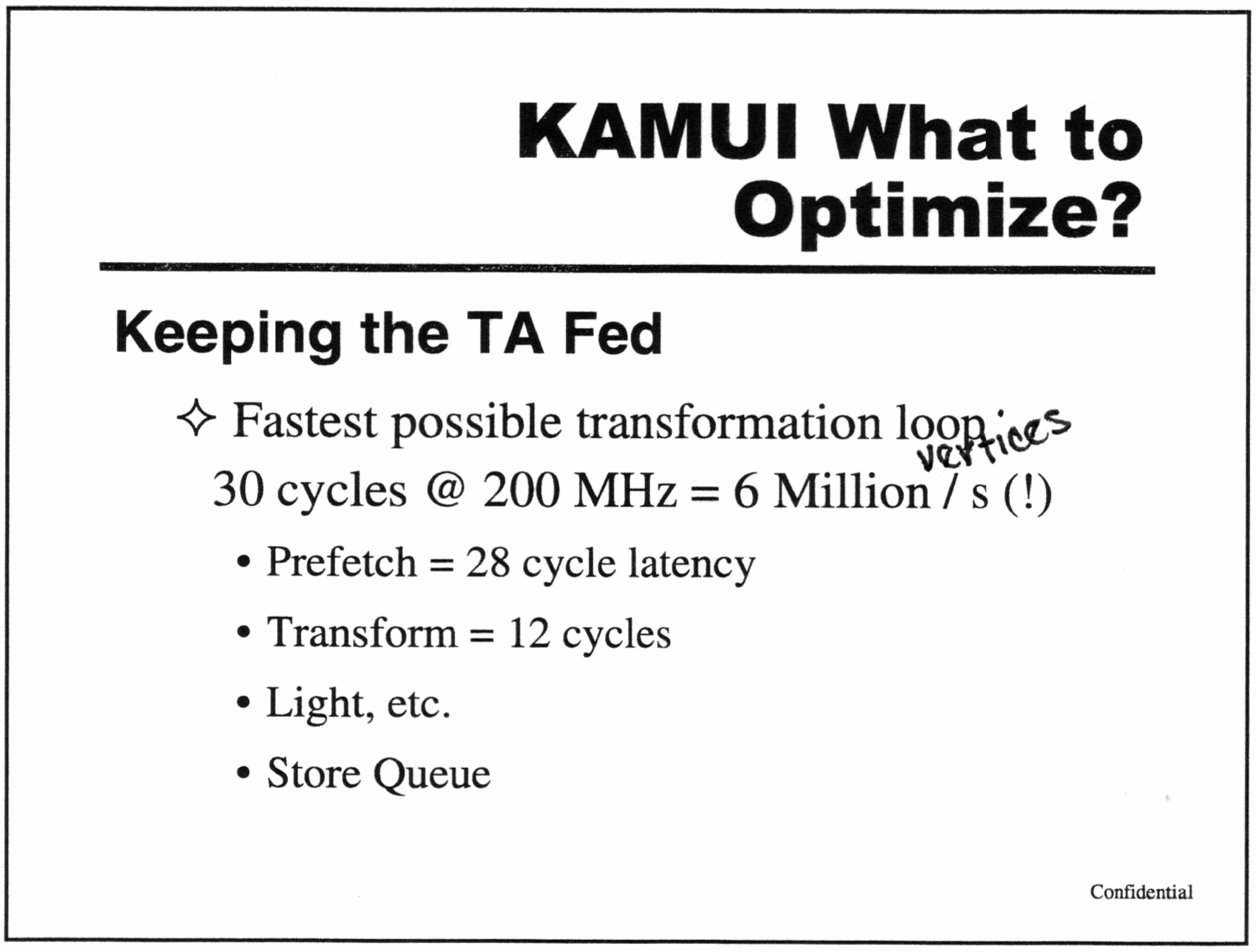


KAMUI What to Optimize?

SH-4 Operand Cache

- ✧ Codescape functional profiler & prfdump
- ✧ Prefetch
  - 28 cycle latency
- ✧ OCRAM mode - use as work area
  - 8 kbyte memory-mapped cache RAM

Confidential

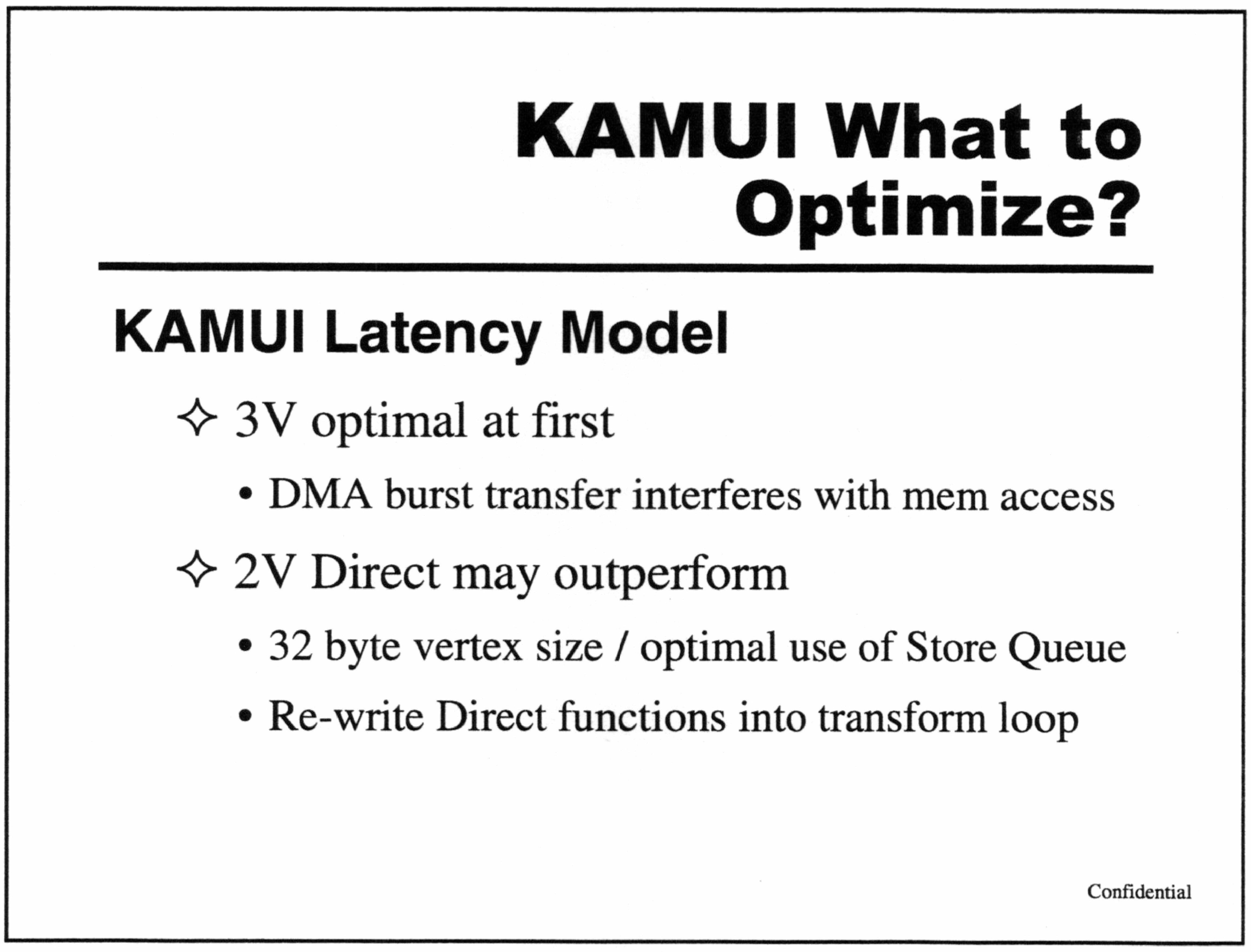


KAMUI What to Optimize?

Keeping the TA Fed

- ✧ Fastest possible transformation loop: 30 cycles @ 200 MHz = 6 Million vertices / s (!)
  - Prefetch = 28 cycle latency
  - Transform = 12 cycles
  - Light, etc.
  - Store Queue

Confidential

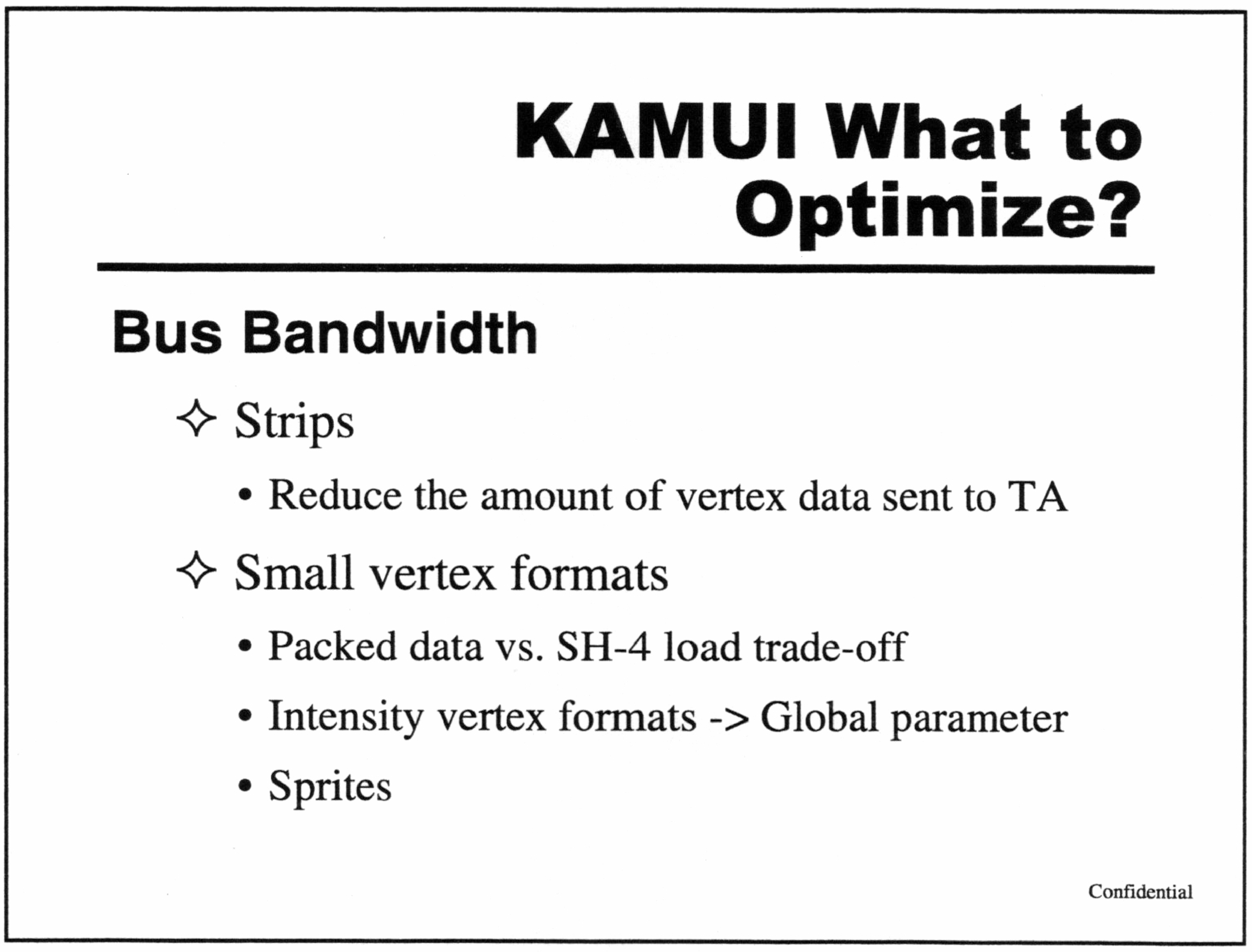


KAMUI What to Optimize?

KAMUI Latency Model

- 3V optimal at first
  - DMA burst transfer interferes with mem access
- 2V Direct may outperform
  - 32 byte vertex size / optimal use of Store Queue
  - Re-write Direct functions into transform loop

Confidential

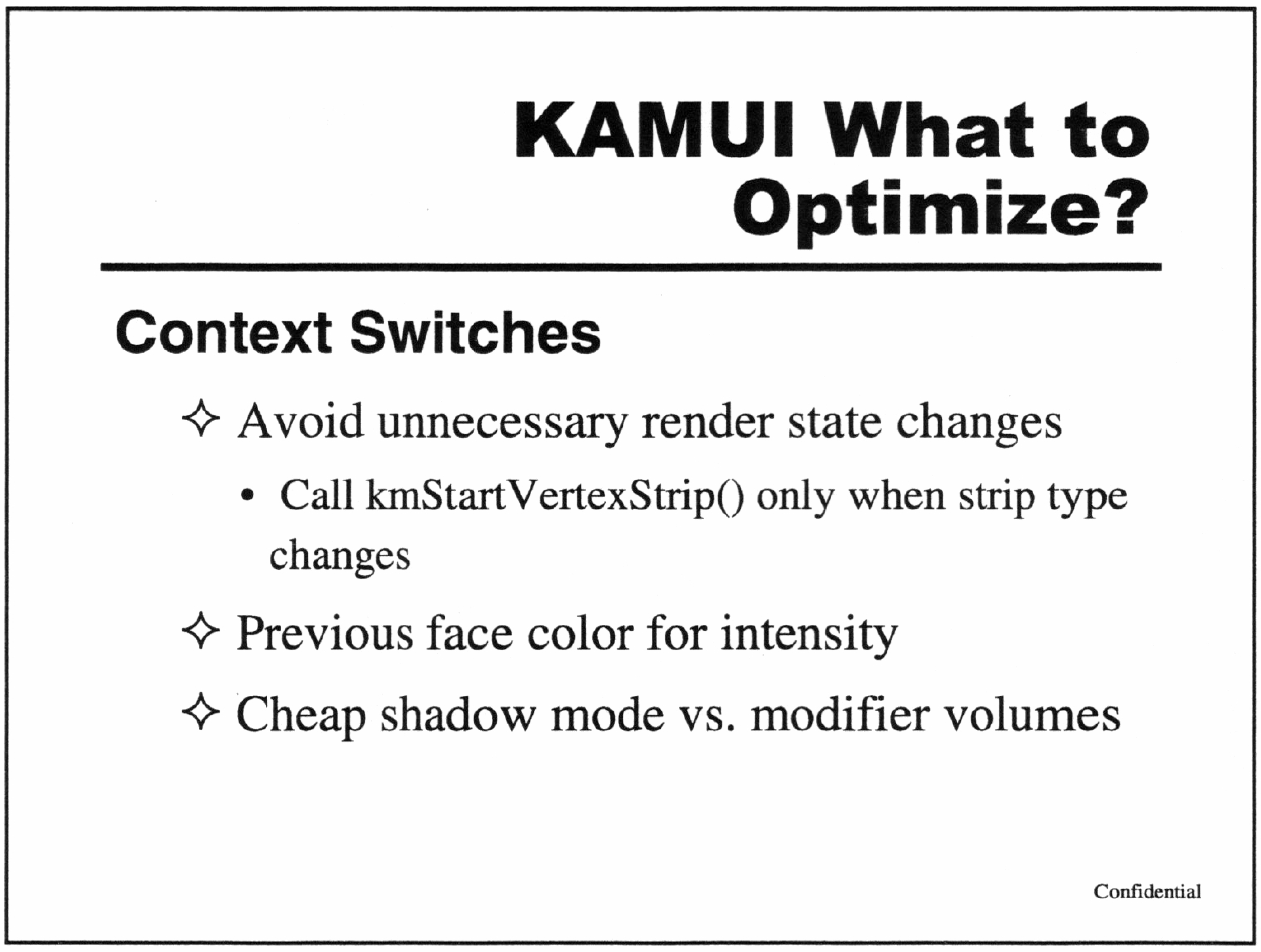


KAMUI What to Optimize?

Bus Bandwidth

- Strips
  - Reduce the amount of vertex data sent to TA
- Small vertex formats
  - Packed data vs. SH-4 load trade-off
  - Intensity vertex formats -> Global parameter
  - Sprites

Confidential

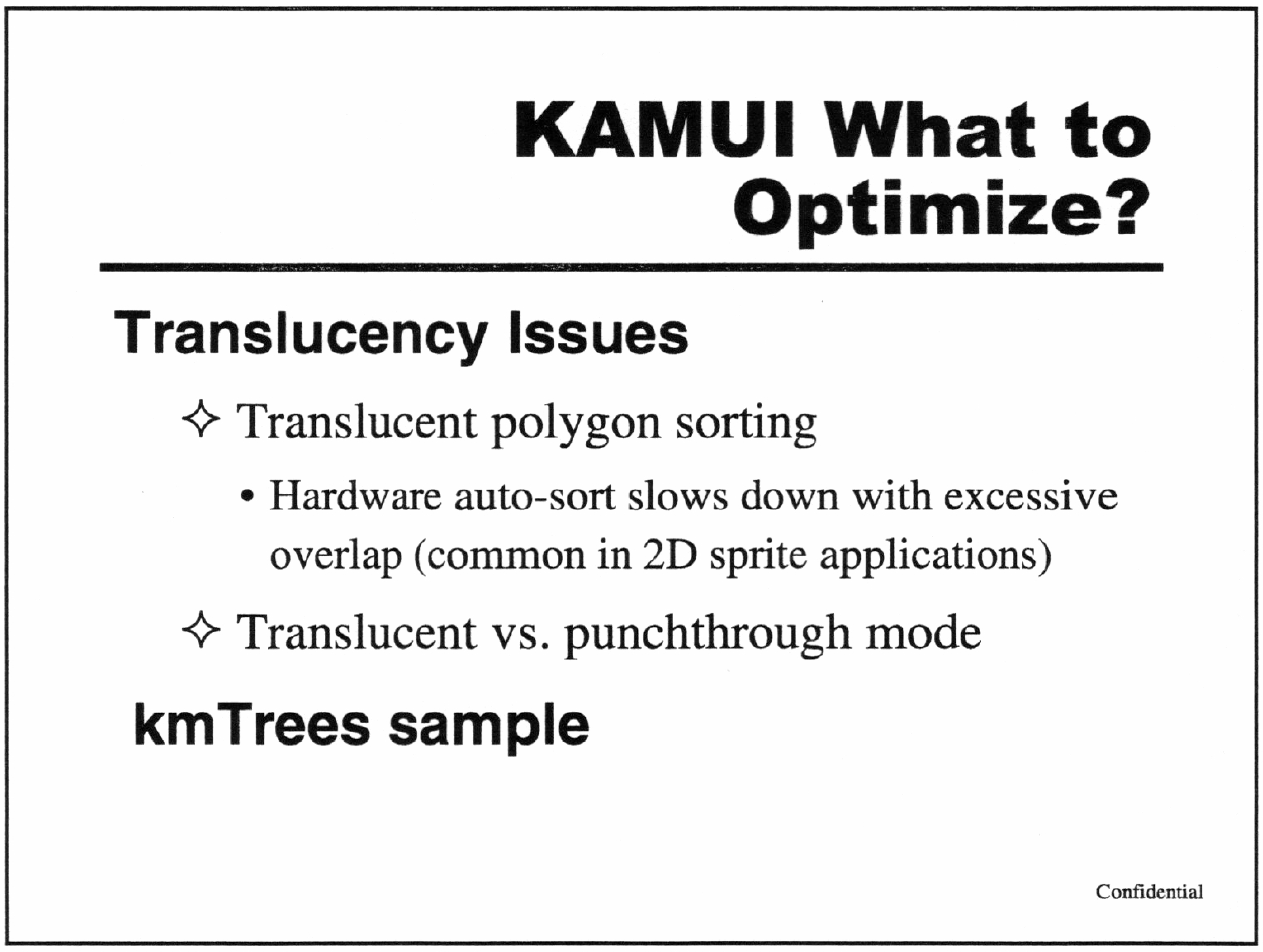


KAMUI What to Optimize?

Context Switches

- ✧ Avoid unnecessary render state changes
  - Call kmStartVertexStrip() only when strip type changes
- ✧ Previous face color for intensity
- ✧ Cheap shadow mode vs. modifier volumes

Confidential



KAMUI What to Optimize?

Translucency Issues

- ✧ Translucent polygon sorting
  - Hardware auto-sort slows down with excessive overlap (common in 2D sprite applications)
- ✧ Translucent vs. punchthrough mode

kmTrees sample

Confidential

KAMUI What to Optimize?

Texture Performance

- ✧ Use mip-maps
  - Guarantees best texture-fit is used
  - Equivalent to smaller textures (2K page size)
- ✧ Use bilinear filtering (64-bit bus)

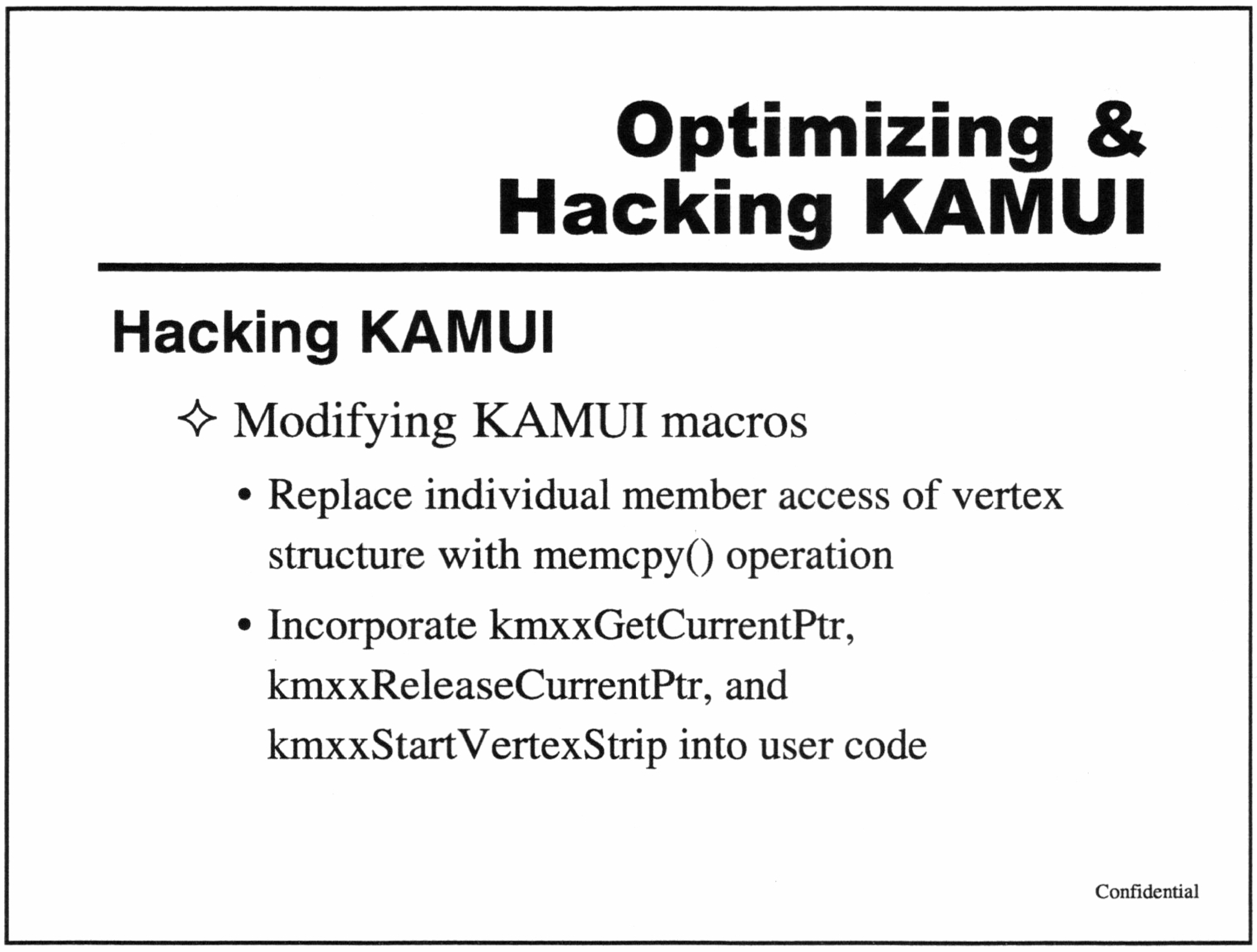
Confidential

KAMUI What to Optimize?

Other Graphics Performance Issues

- ✧ Full-screen anti-aliasing (fill rate)
- ✧ 24-bit mode
- ✧ Trilinear and anisotropic filtering look nice, cost alot
- ✧ Mip-map exact D calculation
- ✧ Internal strip size

Confidential

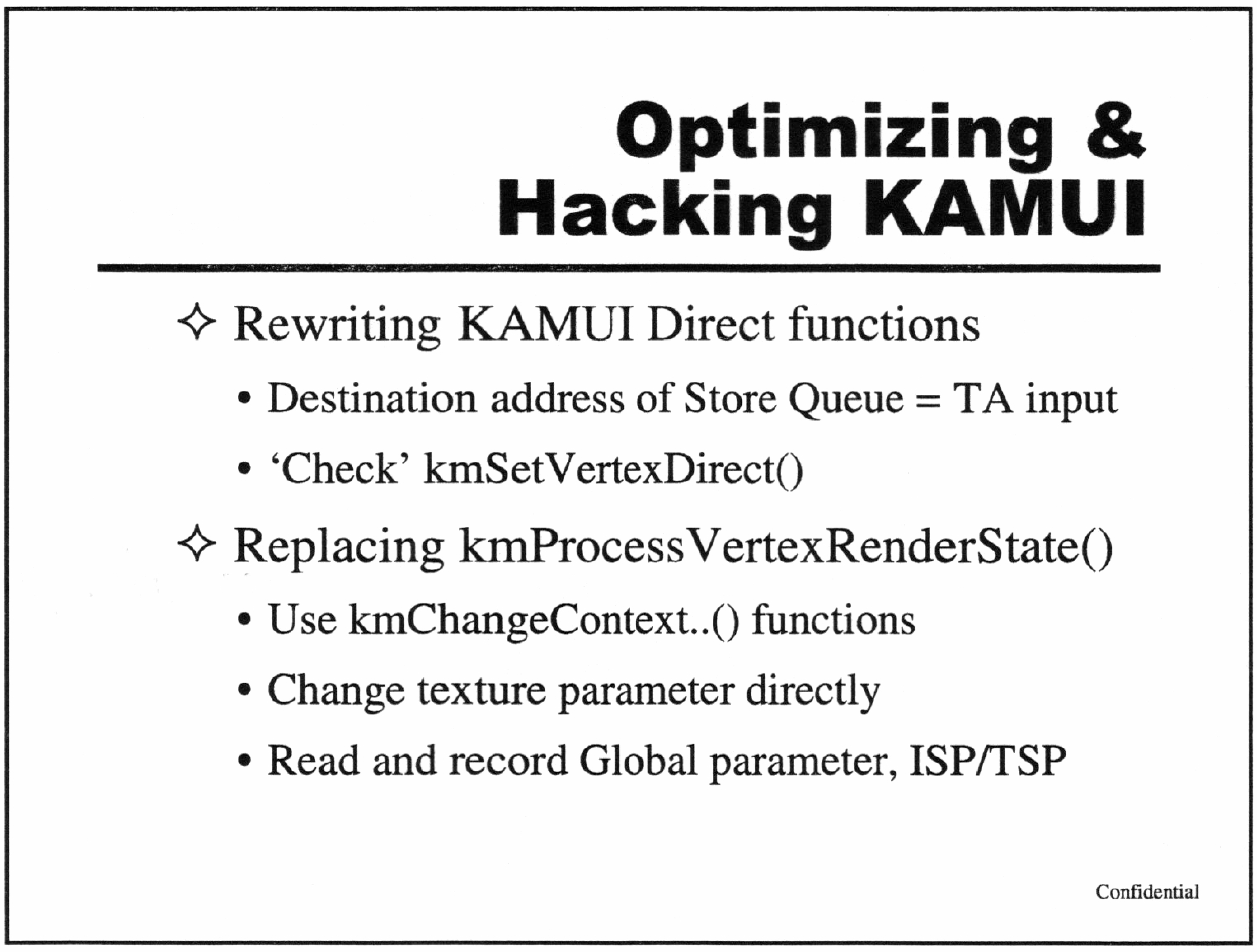


Optimizing & Hacking KAMUI

Hacking KAMUI

- Modifying KAMUI macros
  - Replace individual member access of vertex structure with memcpy() operation
  - Incorporate kmxxGetCurrentPtr, kmxxReleaseCurrentPtr, and kmxxStartVertexStrip into user code

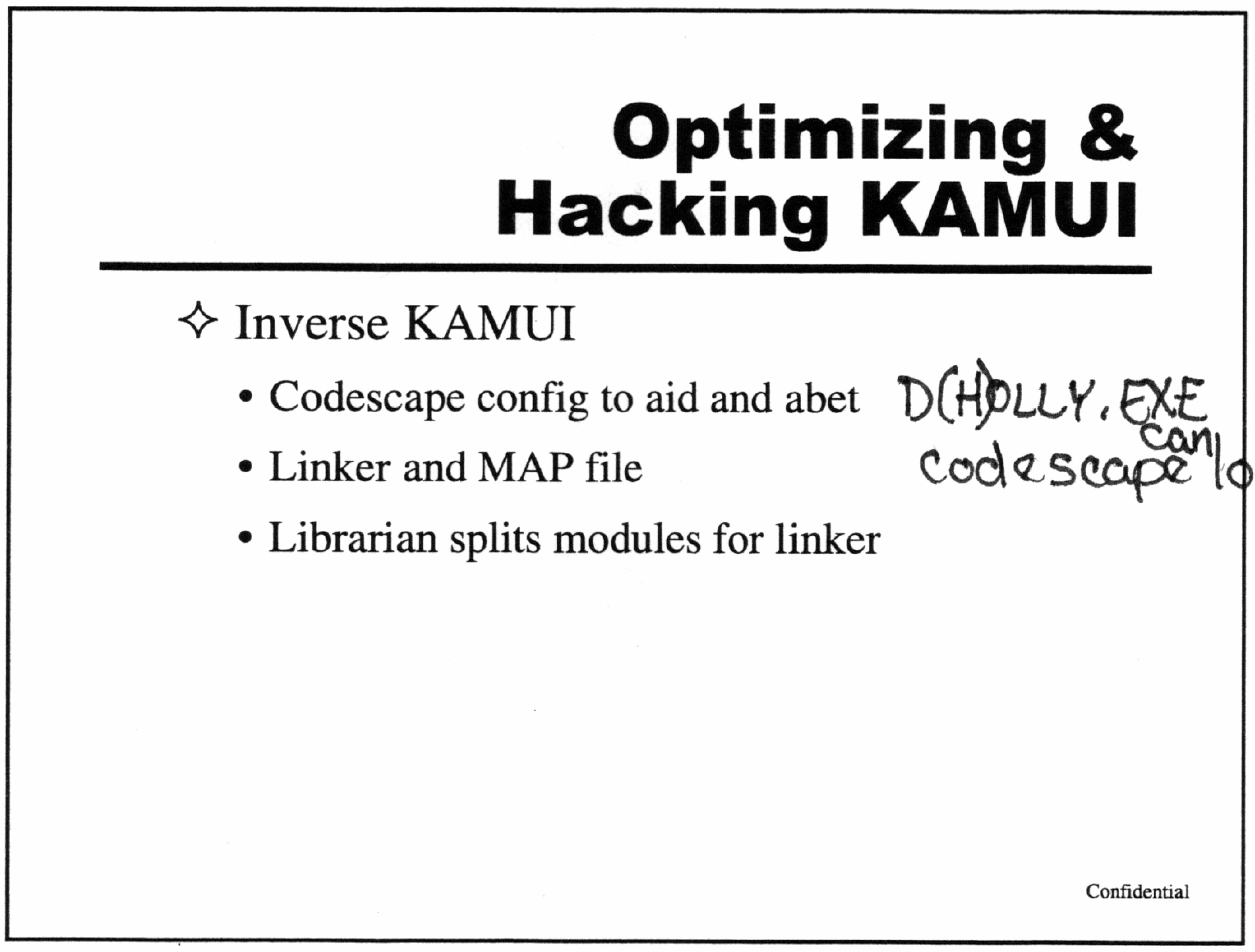
Confidential



Optimizing & Hacking KAMUI

- Rewriting KAMUI Direct functions
  - Destination address of Store Queue = TA input
  - ‘Check’ kmSetVertexDirect()
- Replacing kmProcessVertexRenderState()
  - Use kmChangeContext..() functions
  - Change texture parameter directly
  - Read and record Global parameter, ISP/TSP

Confidential



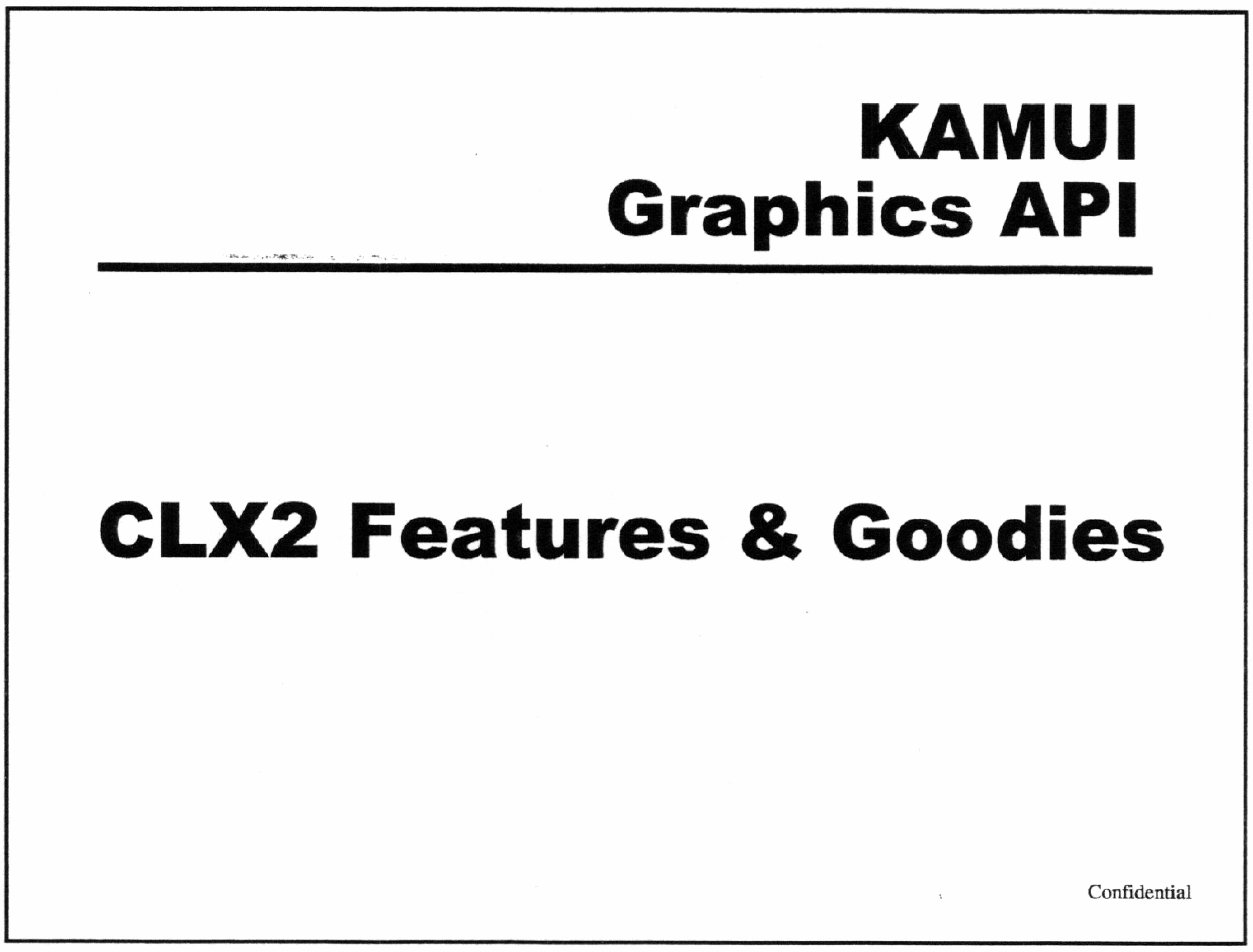
Optimizing & Hacking KAMUI

✧ Inverse KAMUI

- Codescape config to aid and abet
- Linker and MAP file
- Librarian splits modules for linker

Confidential

D(H)OLLY.EXE sets areas that codescape can look at.



KAMUI Graphics API

CLX2 Features & Goodies

Confidential

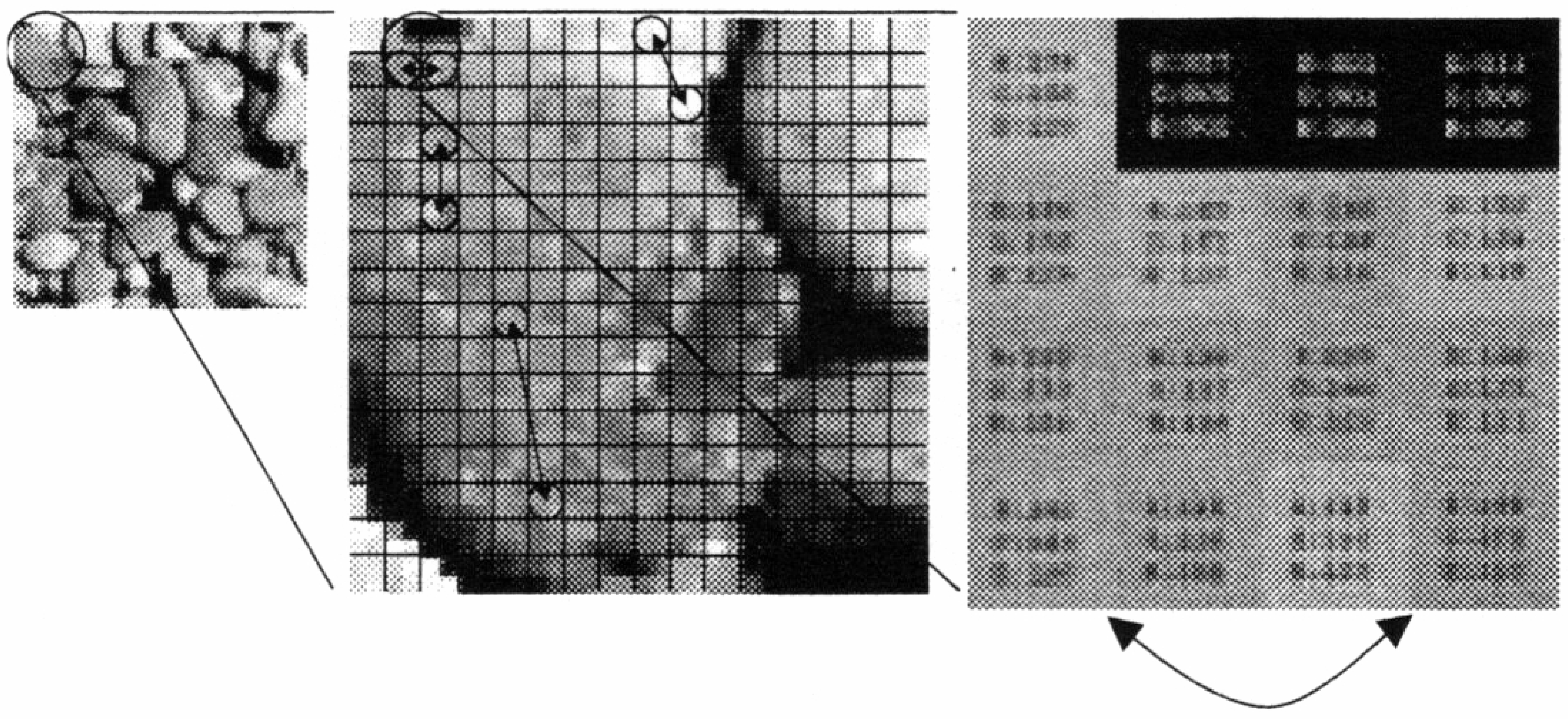
CLX2 Features and Goodies

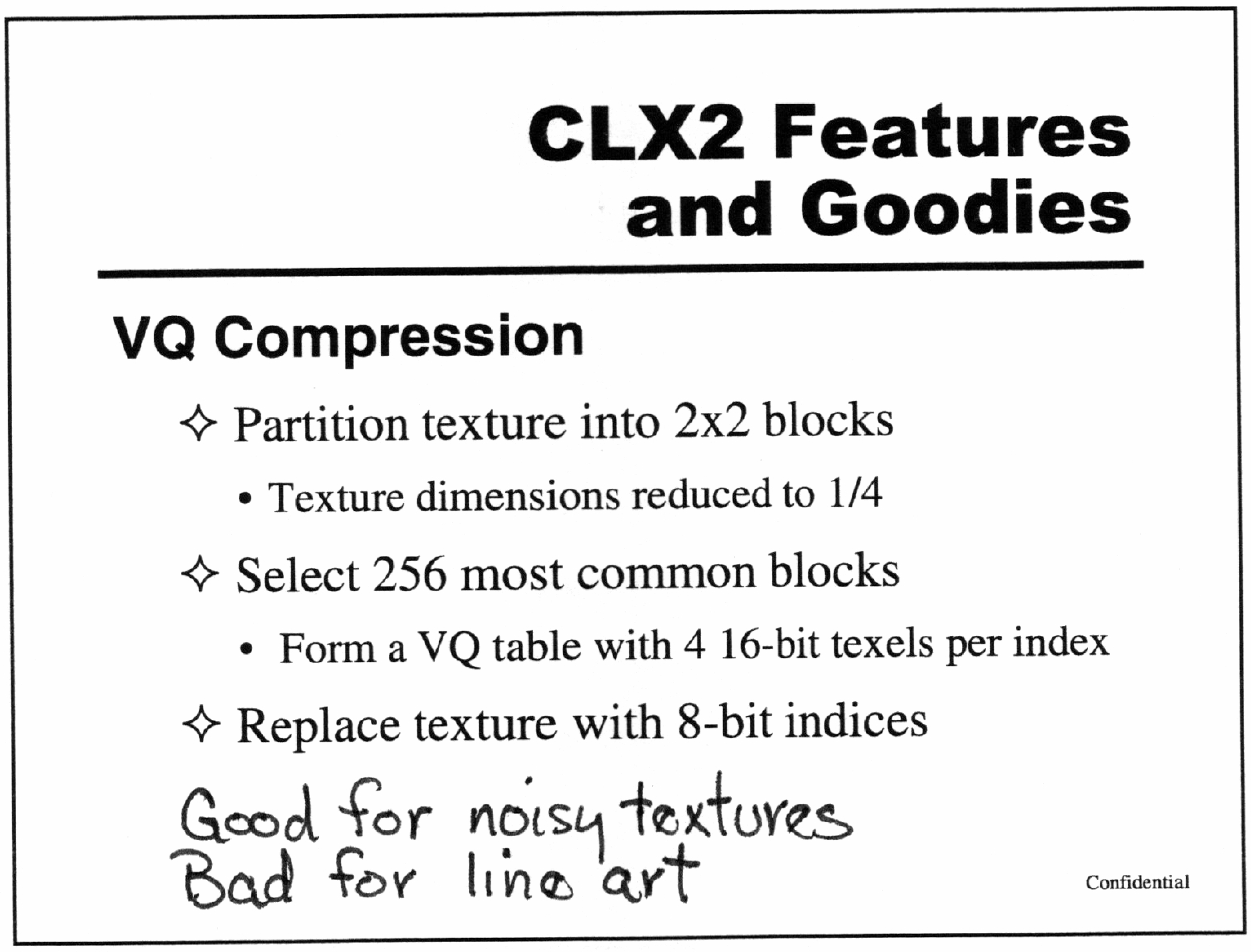
Paletted Textures

- ✧ 1024 entry super-palette (16-bit / 32-bit)
  - 64 banks
- ✧ 4-bit textures, banks 0 - 63
- ✧ 8-bit textures, banks 0, 16, 32, 48
- ✧ Palette updates once-per-scene

Confidential

VQ Compression





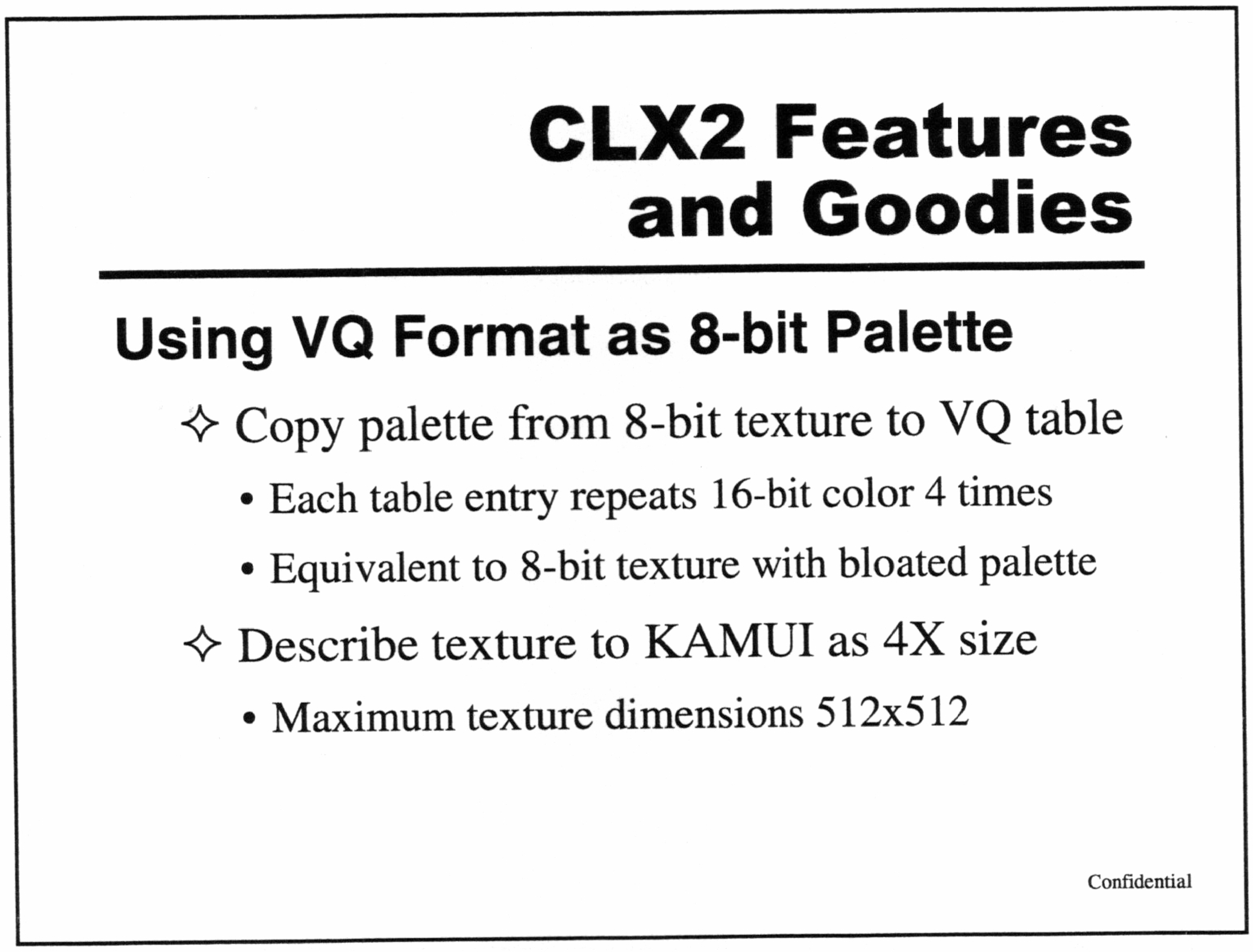
CLX2 Features and Goodies

VQ Compression

- ✧ Partition texture into 2x2 blocks
  - Texture dimensions reduced to 1/4
- ✧ Select 256 most common blocks
  - Form a VQ table with 4 16-bit texels per index
- ✧ Replace texture with 8-bit indices

Good for noisy textures
Bad for line art

Confidential



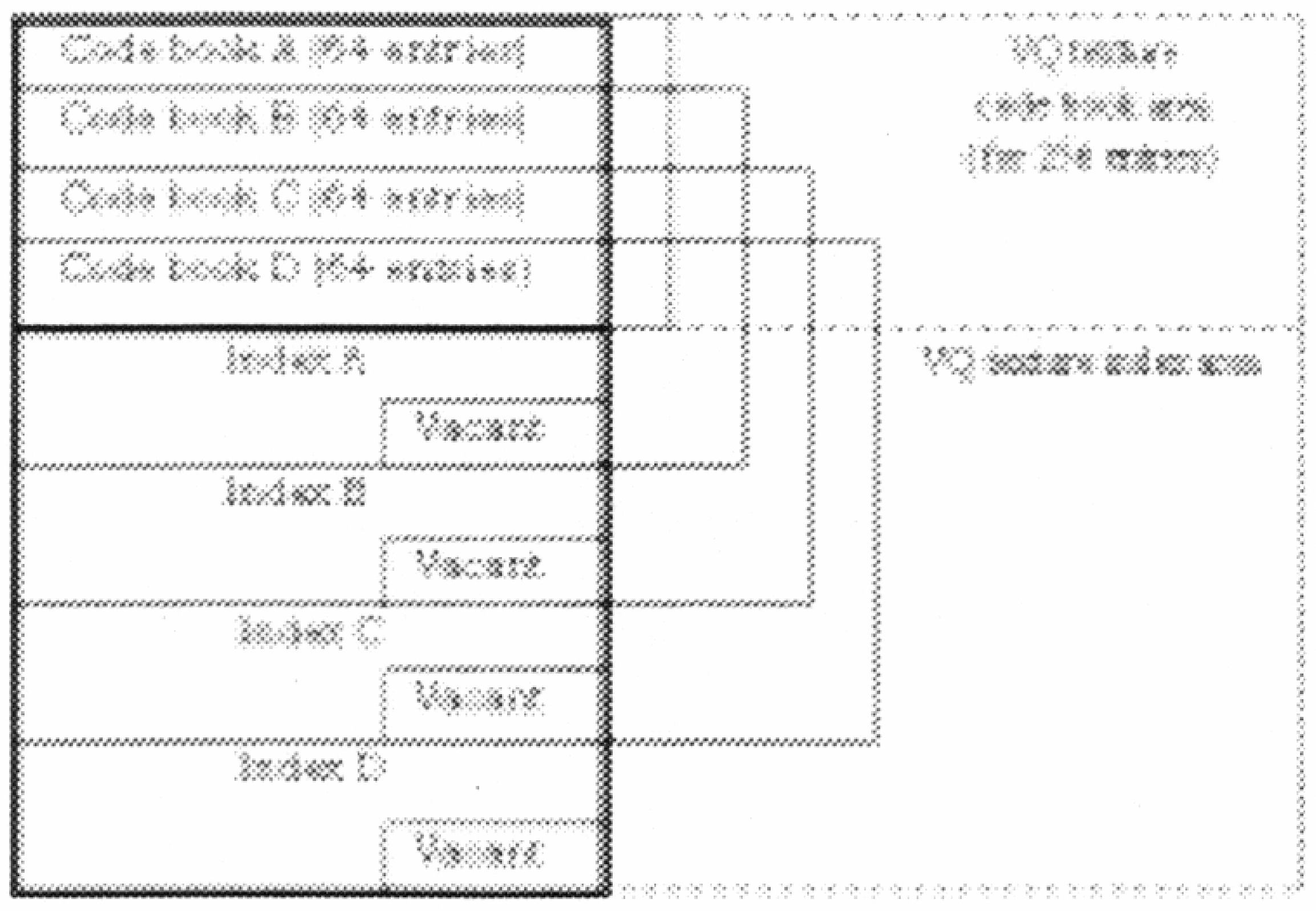
CLX2 Features and Goodies

Using VQ Format as 8-bit Palette

- ✧ Copy palette from 8-bit texture to VQ table
  - Each table entry repeats 16-bit color 4 times
  - Equivalent to 8-bit texture with bloated palette
- ✧ Describe texture to KAMUI as 4X size
  - Maximum texture dimensions 512x512

Confidential

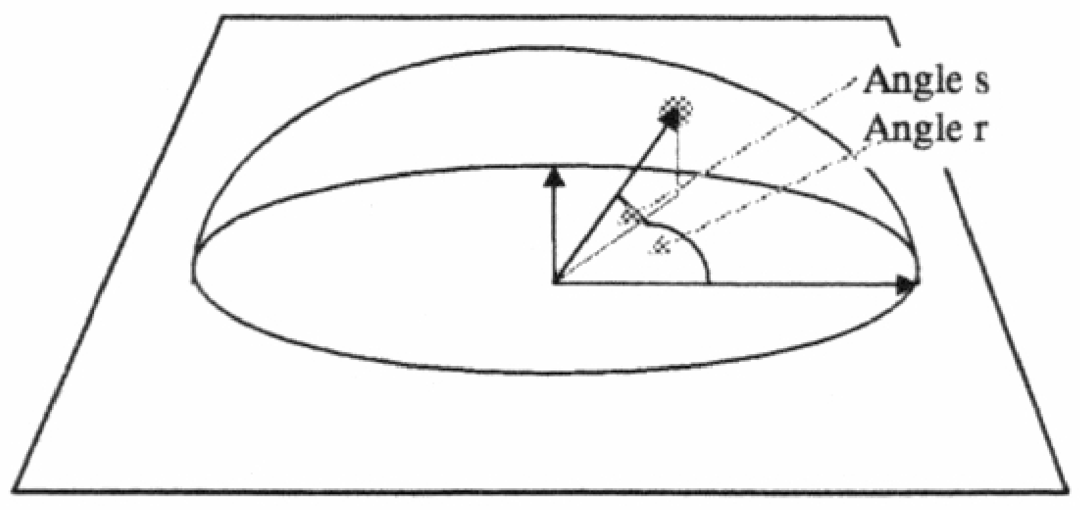
Small VQ Format



Bump Mapping

$$x_s = \cos(s')\cos(r') \qquad s' = \pi/_2 \frac{s}{256}$$
$$y_s = \sin(s') \qquad \text{where}$$
$$z_s = \cos(s')\sin(r') \qquad r' = 2\pi \frac{r}{256}$$

8-bit s and r values stored in texture

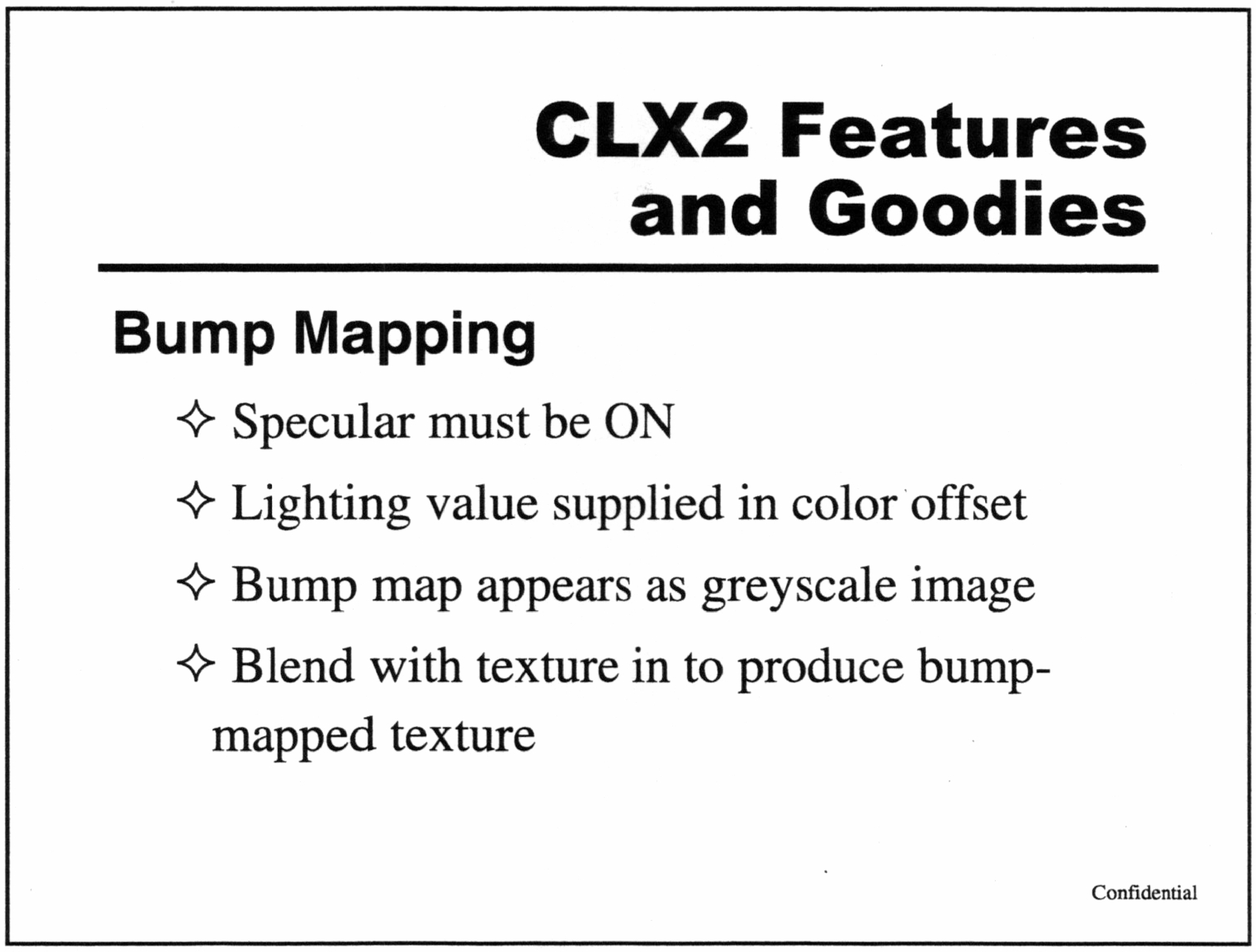


Lighting

$$x^l = \cos(t')\cos(q') \qquad \text{where} \qquad t' = \pi/_2 \frac{t}{256}$$
$$y^l = \sin(t')$$
[illegible]
$$q' = 2\pi \frac{q}{}$$

[illegible]

8-bit k_1, k_2, k_3, and q' values supplied as offset color

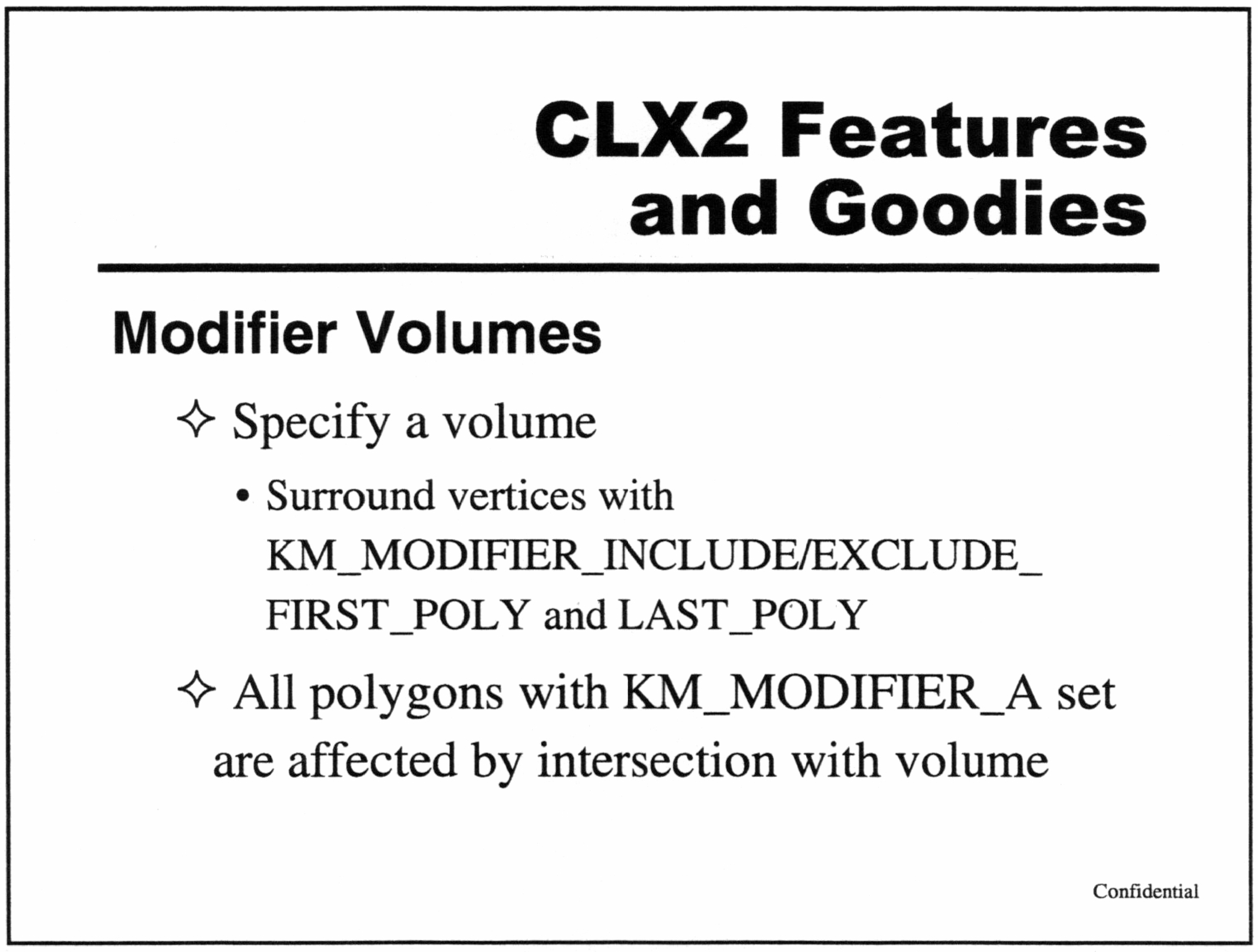


CLX2 Features and Goodies

Bump Mapping

- ✧ Specular must be ON
- ✧ Lighting value supplied in color offset
- ✧ Bump map appears as greyscale image
- ✧ Blend with texture in to produce bump-mapped texture

Confidential



CLX2 Features and Goodies

Modifier Volumes

- ✧ Specify a volume
  - Surround vertices with KM_MODIFIER_INCLUDE/EXCLUDE_ FIRST_POLY and LAST_POLY
- ✧ All polygons with KM_MODIFIER_A set are affected by intersection with volume

Confidential

CLX2 Features and Goodies

Cheap Shadow Mode

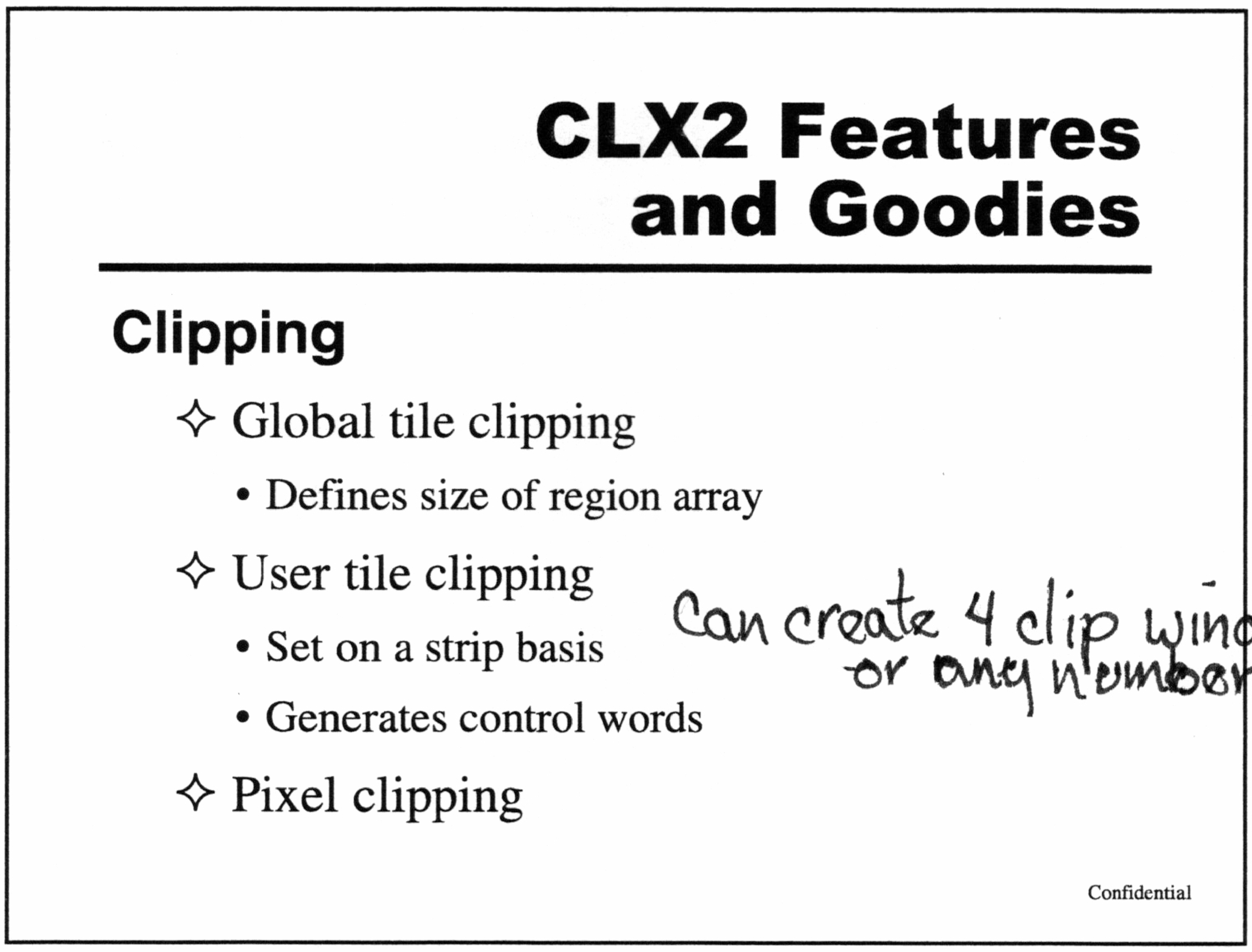
- ✧ Special case of modifier volume
  - All polygons with KM_MODIFIER_A set intersecting shadow volume have luminance modified
  - No need to specify 2 parameter polygons

Confidential

CLX2 Features and Goodies

- ✧ kmSetCheapShadowMode
  - Specify 8-bit intensity value 0 - 255
  - Intensity of -1 turns off
- ✧ Turn on/off cheap shadows surrounding kmProcessVertexRenderState()

Confidential



CLX2 Features and Goodies

Clipping

✧ Global tile clipping

- Defines size of region array

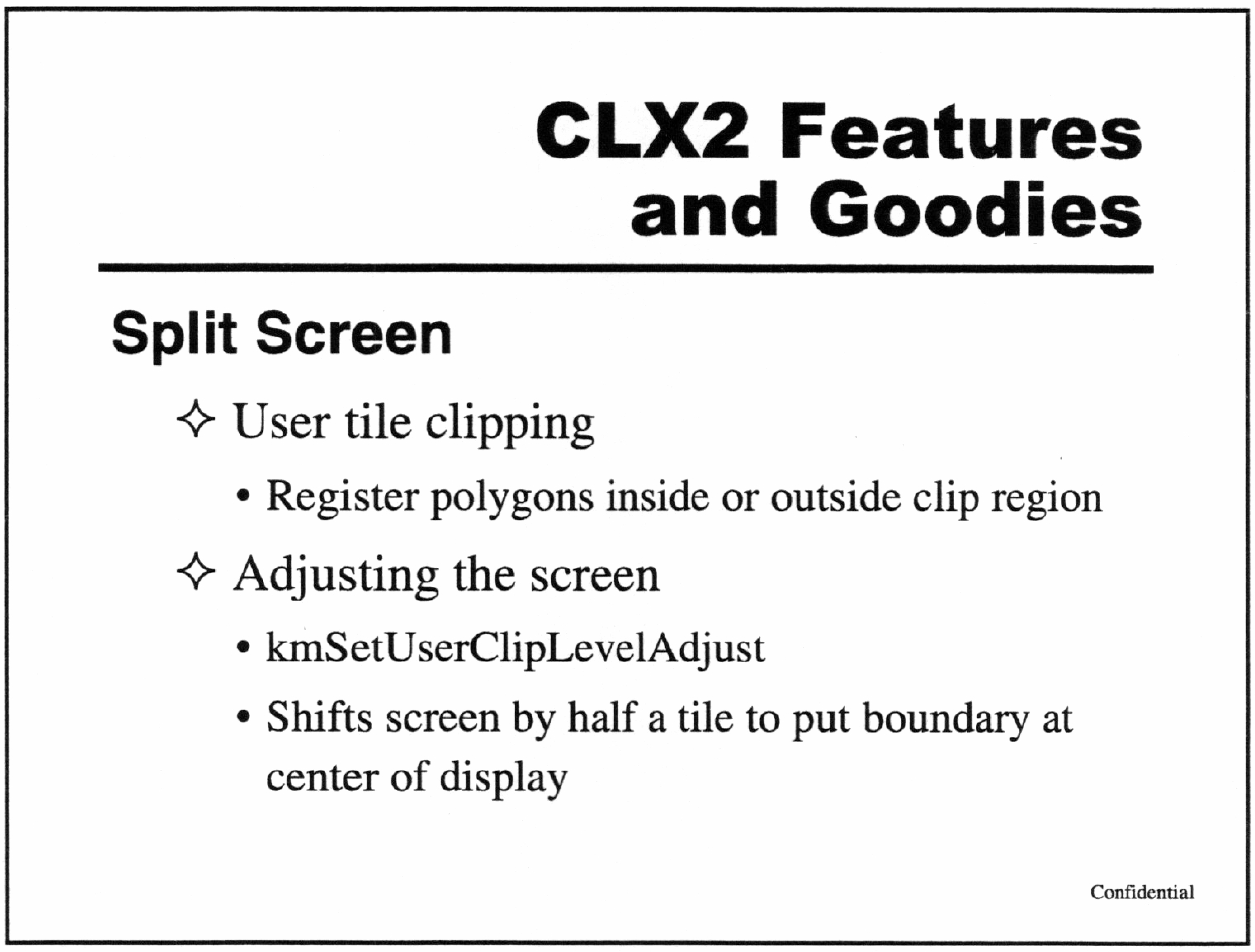
✧ User tile clipping

- Set on a strip basis
- Generates control words

✧ Pixel clipping

Can create 4 clip windows or any number

Confidential



CLX2 Features and Goodies

Split Screen

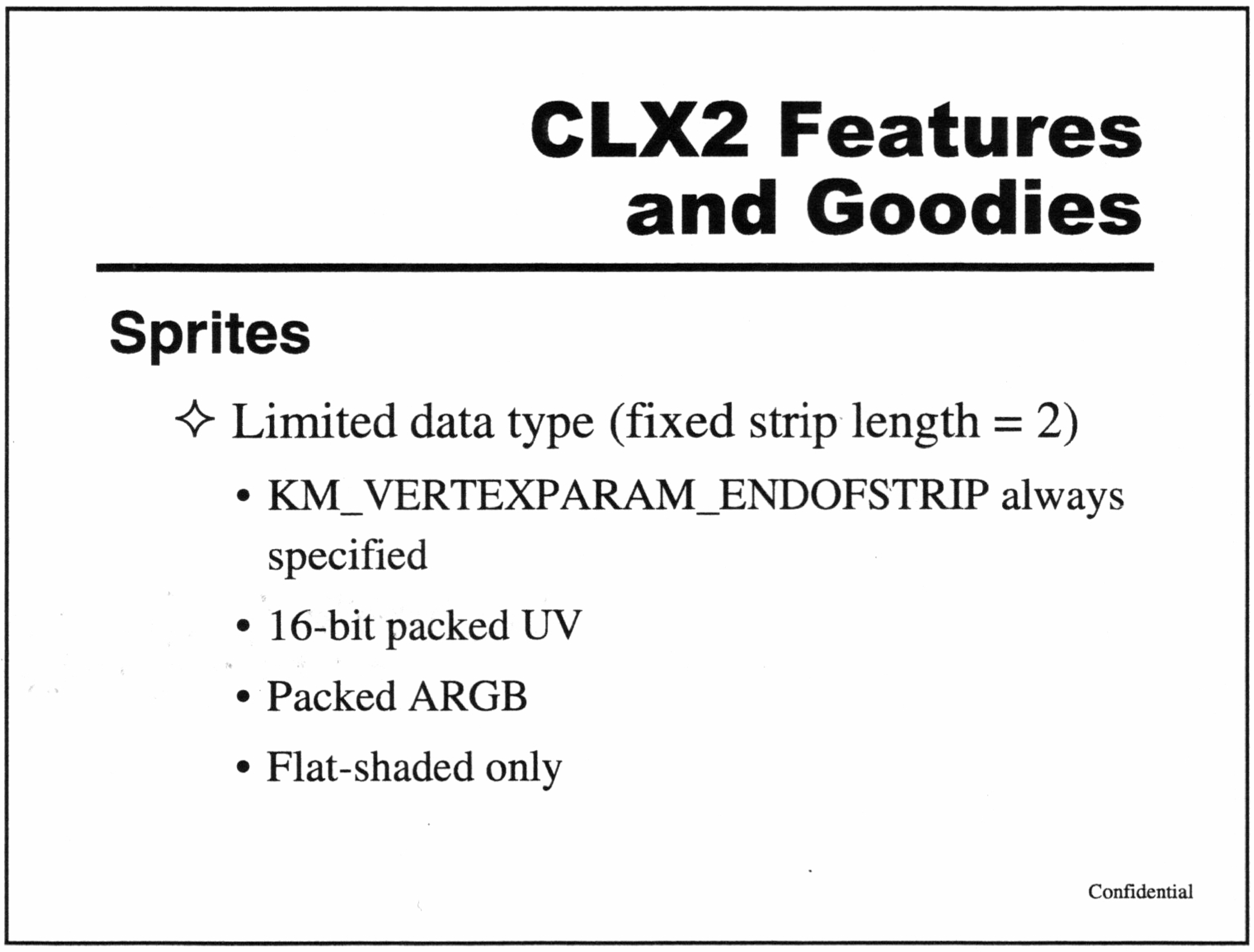
✧ User tile clipping

- Register polygons inside or outside clip region

✧ Adjusting the screen

- kmSetUserClipLevelAdjust
- Shifts screen by half a tile to put boundary at center of display

Confidential

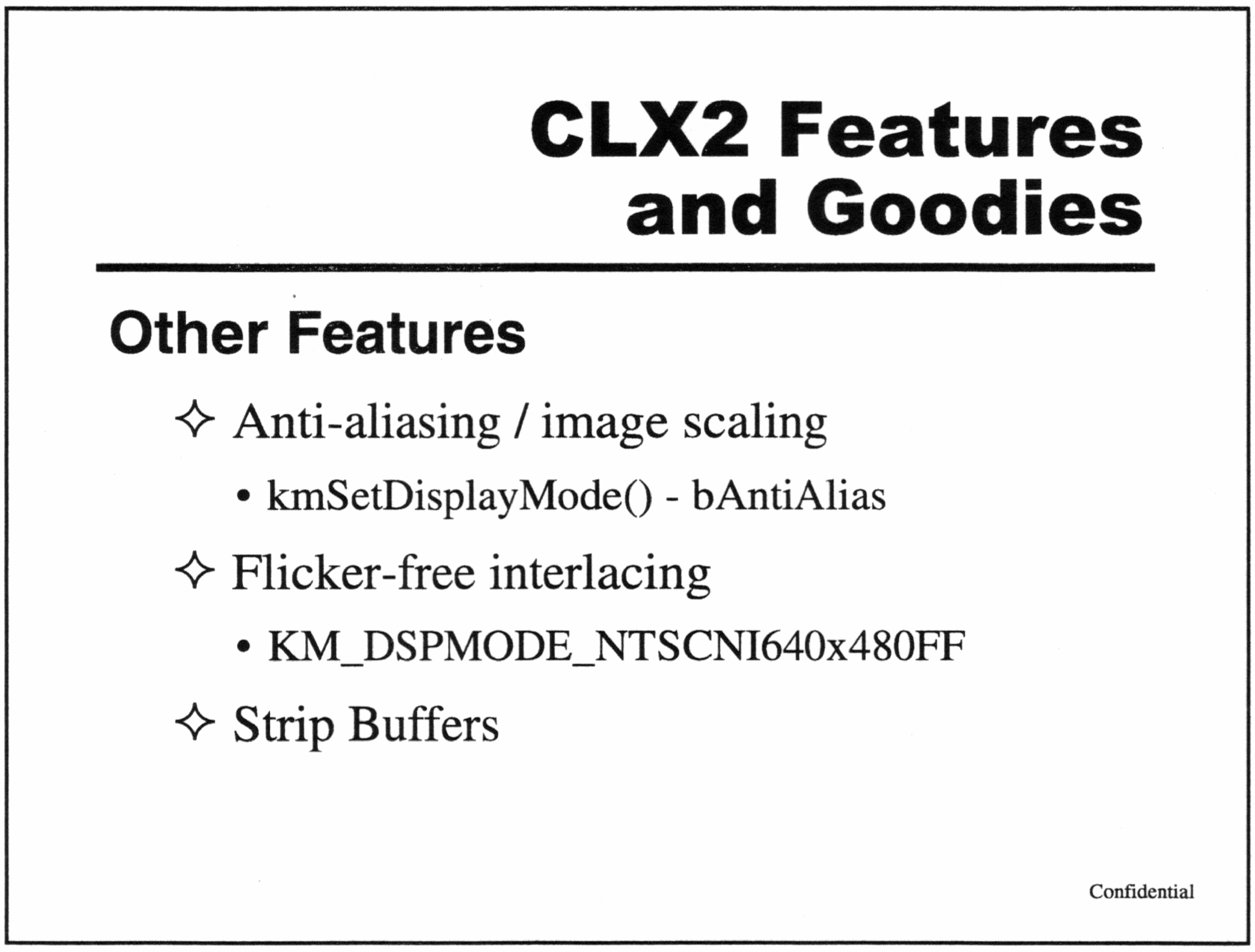


CLX2 Features and Goodies

Sprites

- ✧ Limited data type (fixed strip length = 2)
  - KM_VERTEXPARAM_ENDOFSTRIP always specified
  - 16-bit packed UV
  - Packed ARGB
  - Flat-shaded only

Confidential



CLX2 Features and Goodies

Other Features

- ✧ Anti-aliasing / image scaling
  - kmSetDisplayMode() - bAntiAlias
- ✧ Flicker-free interlacing
  - KM_DSPMODE_NTSCNI640x480FF
- ✧ Strip Buffers

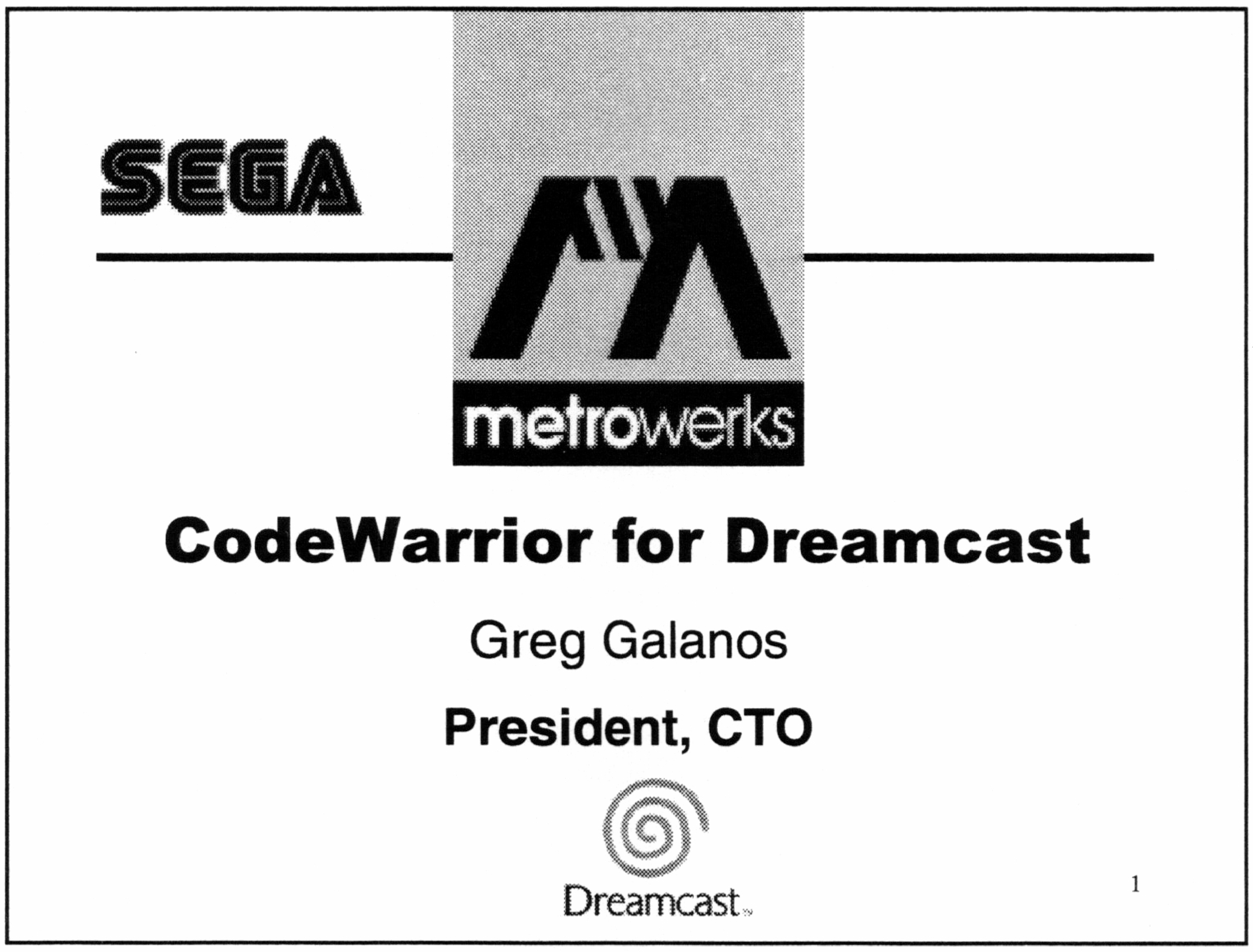
Confidential

SEGA

CodeWarrior for Dreamcast

Greg Galanos
President, CTO
Metrowerks

Dreamcast™



SEGA

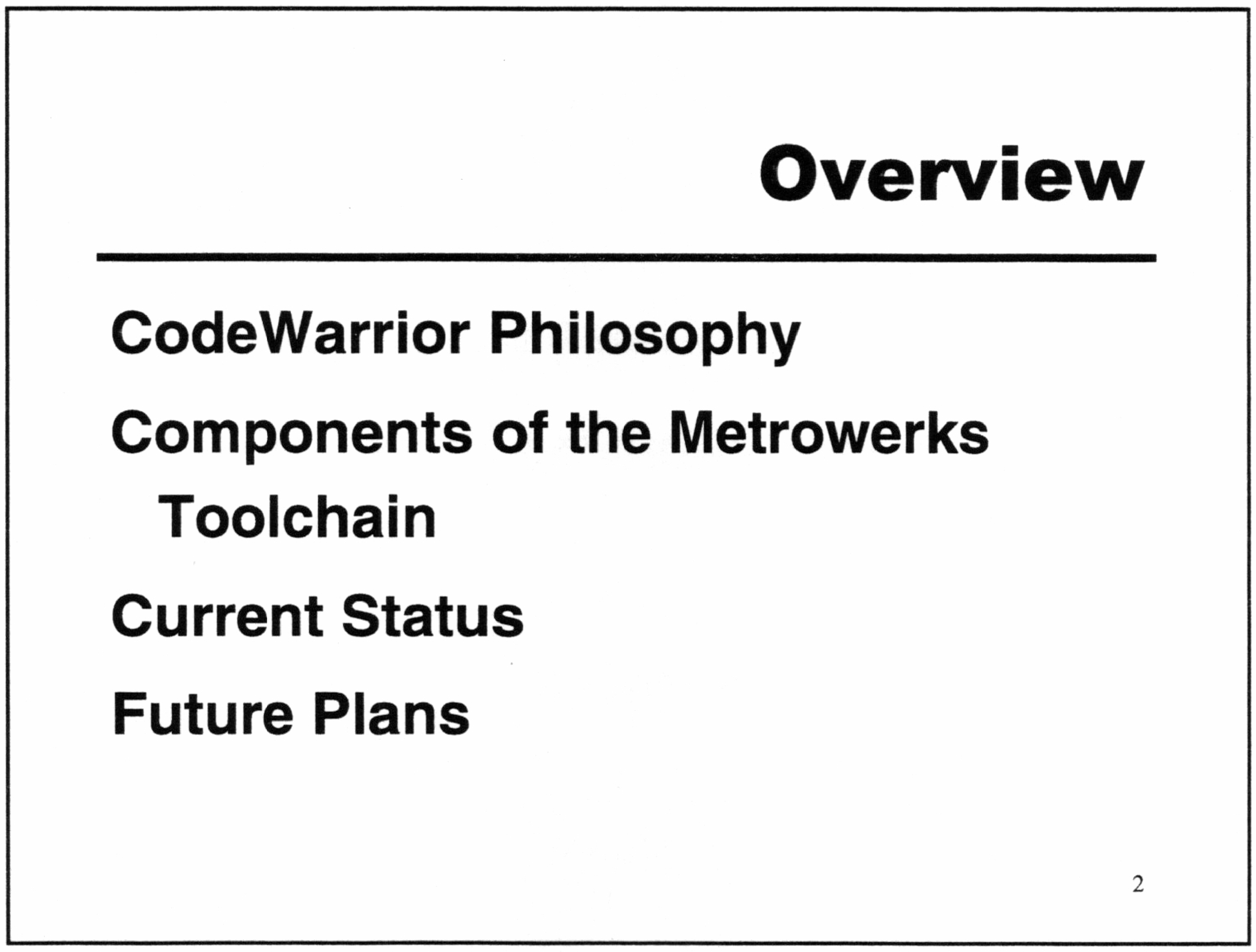
metrowerks

CodeWarrior for Dreamcast

Greg Galanos

President, CTO

Dreamcast



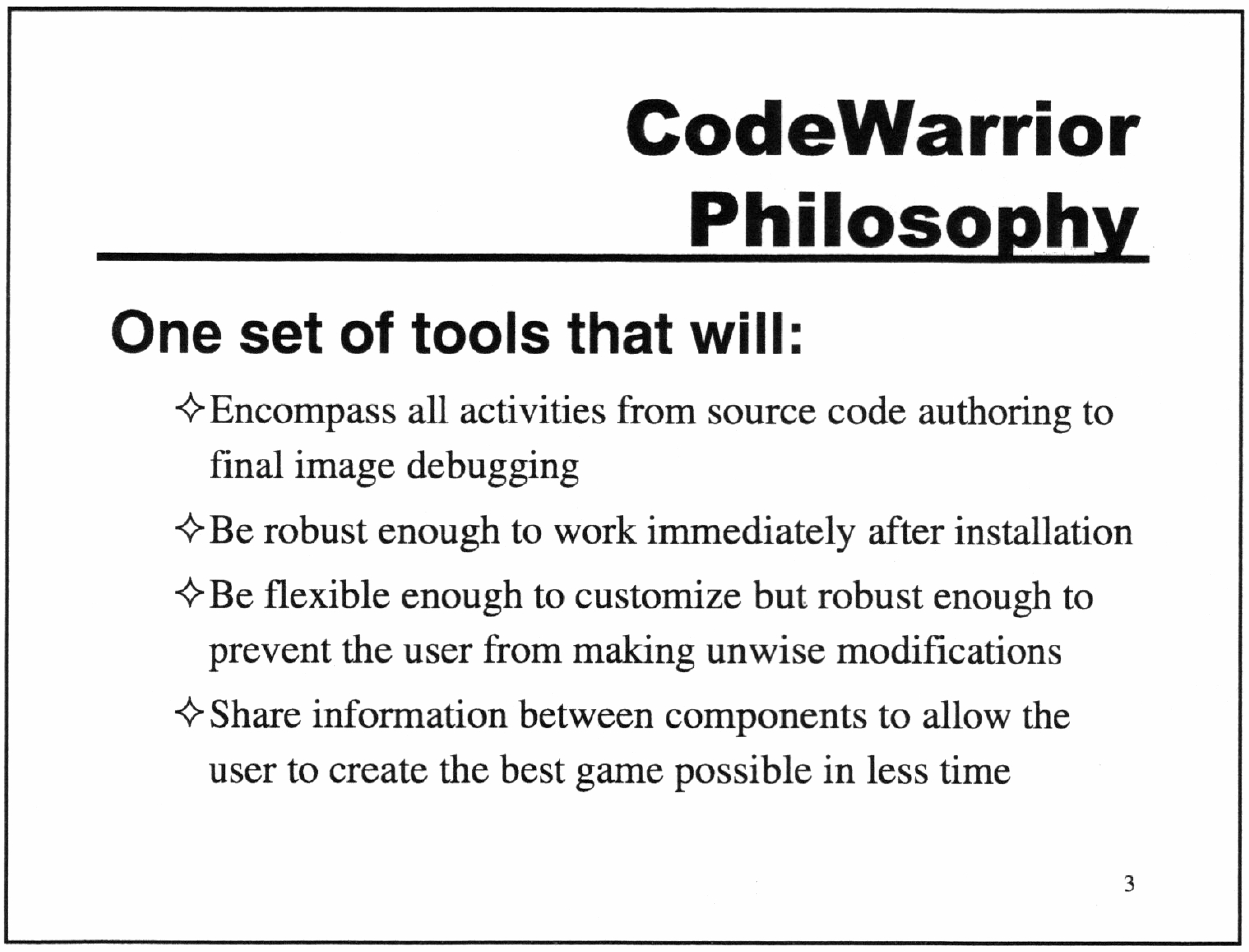
Overview

CodeWarrior Philosophy

Components of the Metrowerks Toolchain

Current Status

Future Plans



CodeWarrior Philosophy

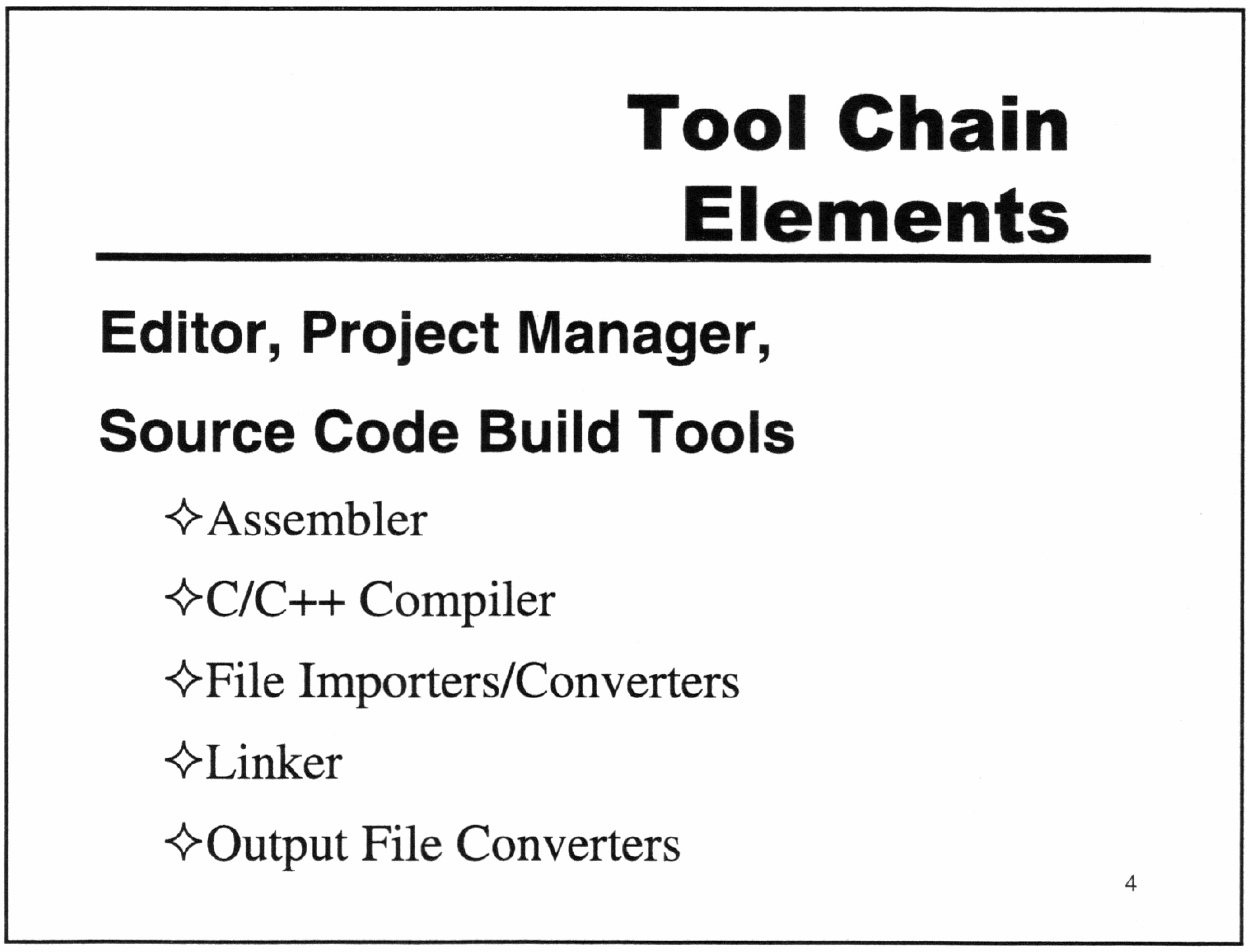
One set of tools that will:

✧Encompass all activities from source code authoring to final image debugging

✧Be robust enough to work immediately after installation

✧Be flexible enough to customize but robust enough to prevent the user from making unwise modifications

✧Share information between components to allow the user to create the best game possible in less time



Tool Chain Elements

Editor, Project Manager,

Source Code Build Tools

✧Assembler

✧C/C++ Compiler

✧File Importers/Converters

✧Linker

✧Output File Converters

Tool Chain Elements - 2

Debugger and Debugging Helper Applications

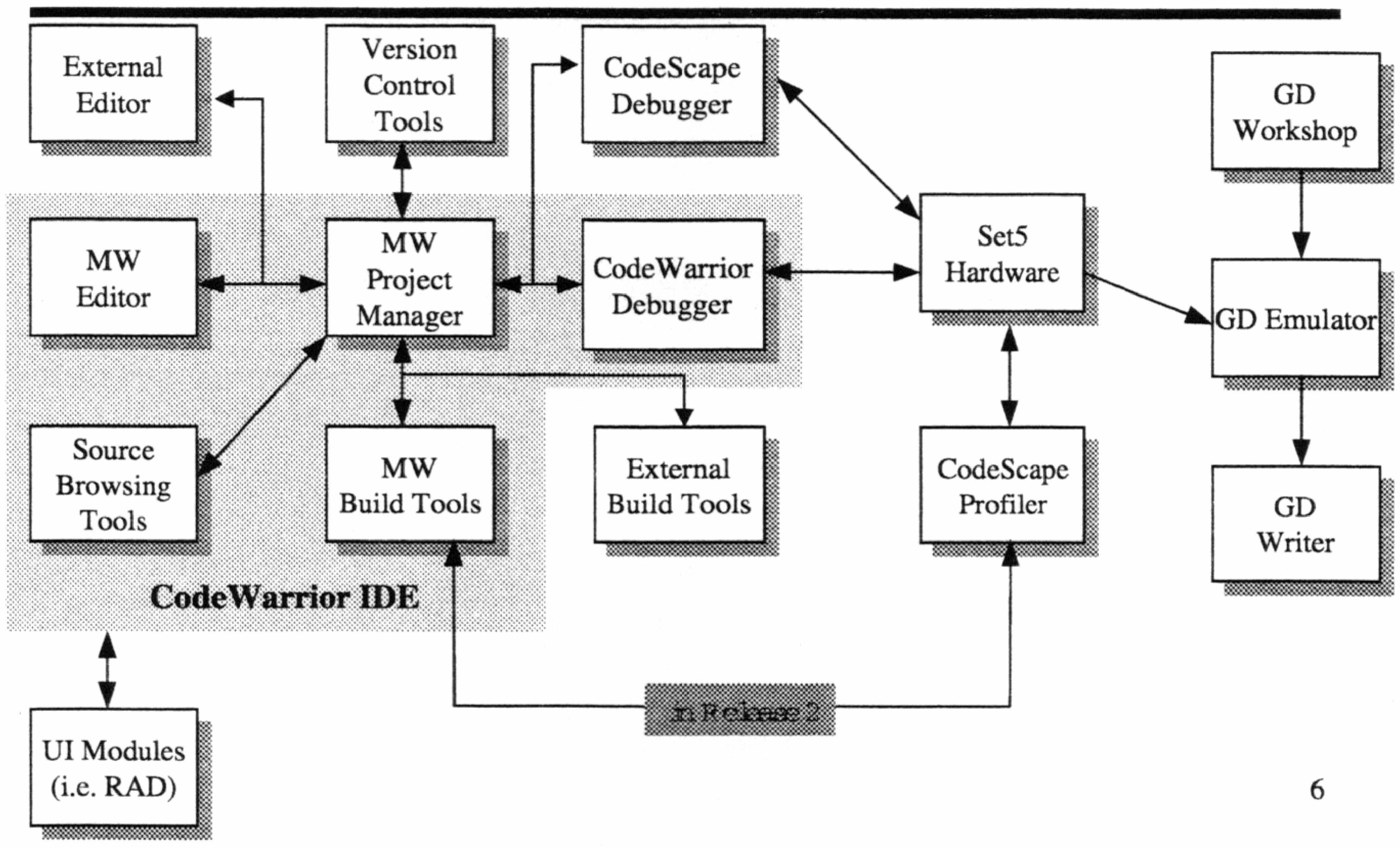
- ✧Metrowerks Debugger and Cross Products CodeScape

GD Emulator, GD Writer

Code Analysis Tools

- ✧Profiler in CodeScape

Tool Chain Flow



External Editor

Version Control Tools

CodeScape Debugger

GD Workshop

MW Editor

MW Project Manager

CodeWarrior Debugger

Set5 Hardware

GD Emulator

Source Browsing Tools

MW Build Tools

External Build Tools

CodeScape Profiler

GD Writer

CodeWarrior IDE

in Release 2

UI Modules (i.e. RAD)

Tool Chain Integration

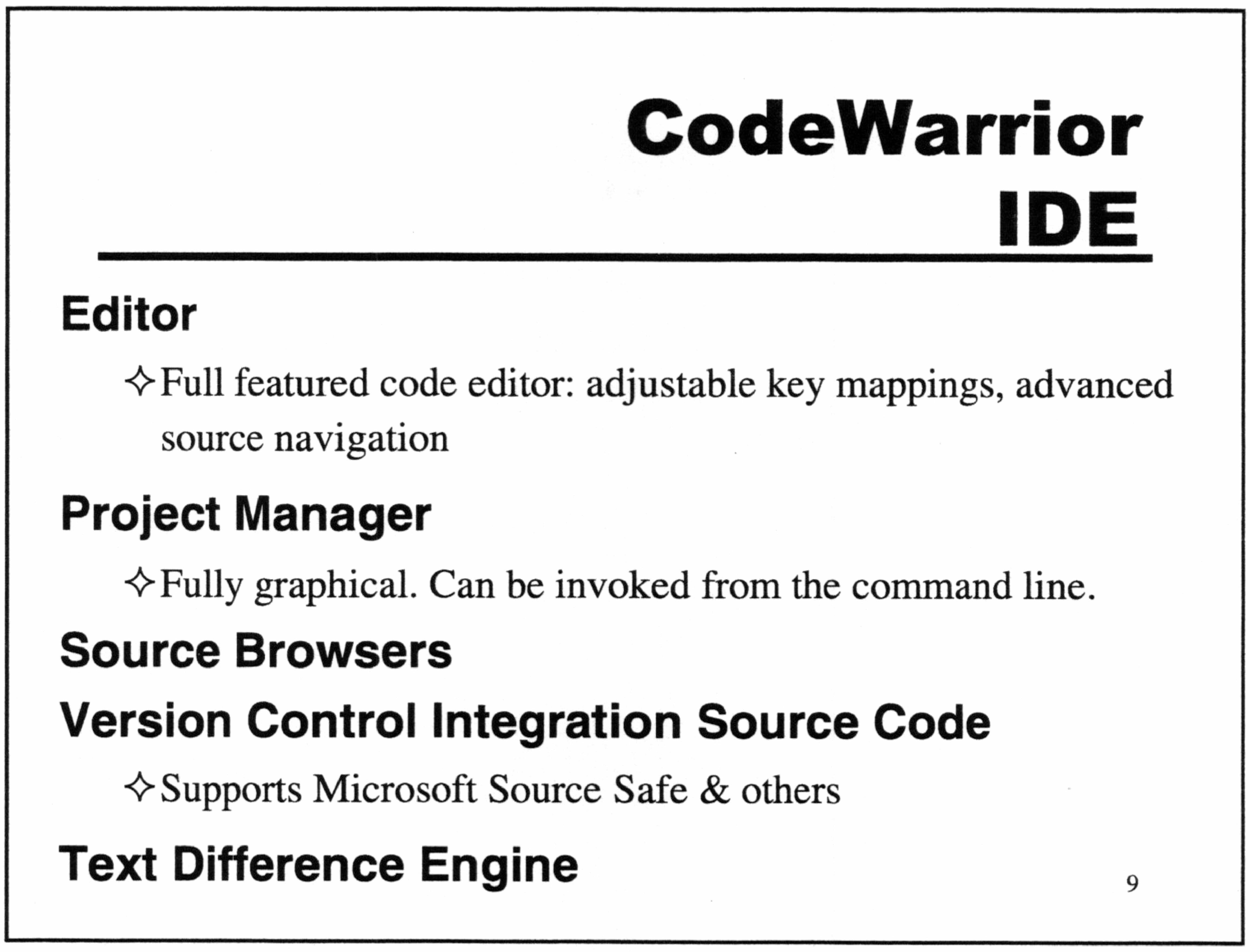
Sharing information and control between logical components

- ✧ One keystroke will cause IDE to invoke the code build tools, launch the debugger, download the application to the hardware, and run it.

Sharing Information

- ✧ Compiler generates information that is used by the code browser. This allows the editor to perform symbol completion, navigate to symbol definition location, jump to a header file that is included by a source file, etc..

Metrowerks Components



CodeWarrior IDE

Editor

- ✧ Full featured code editor: adjustable key mappings, advanced source navigation

Project Manager

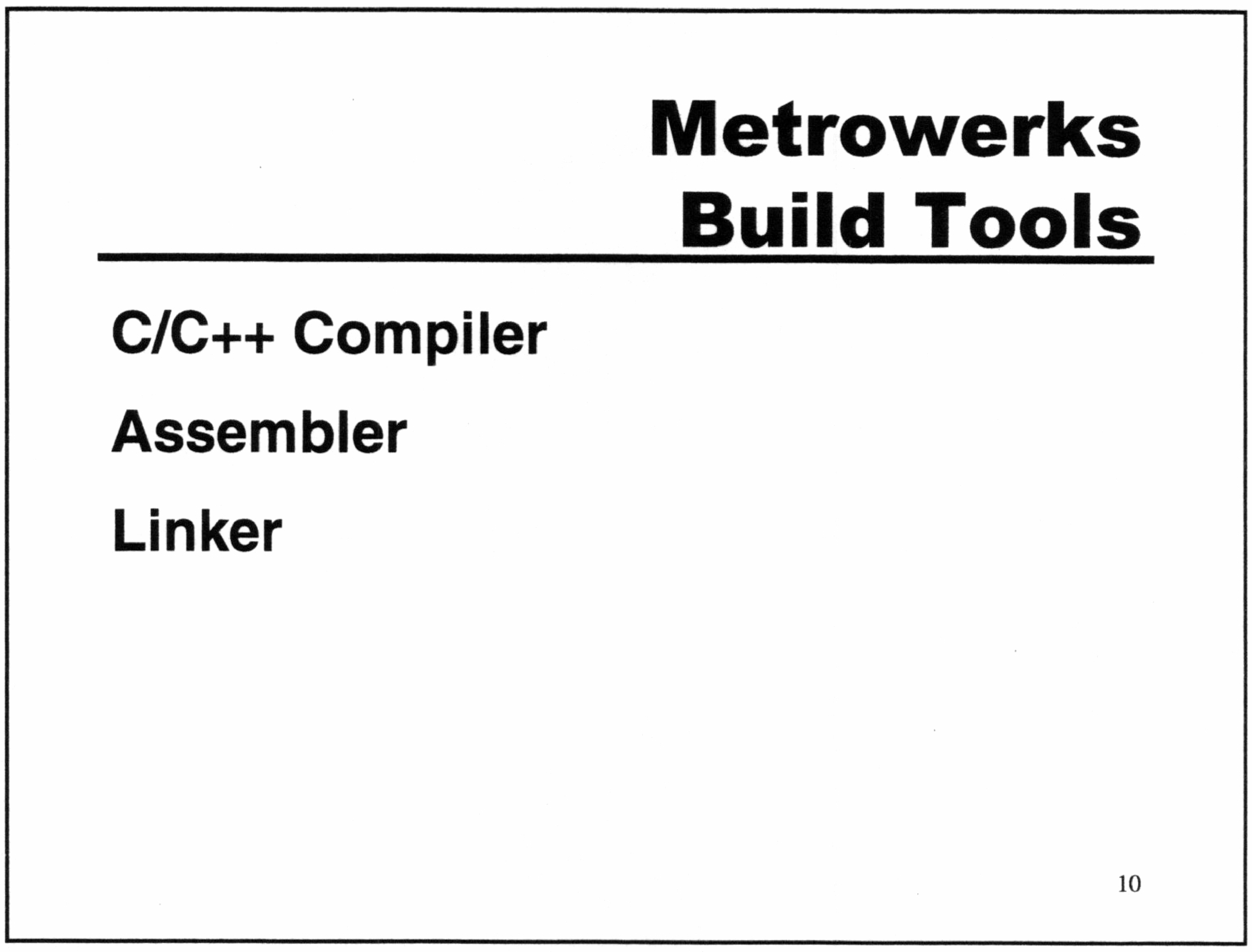
- ✧ Fully graphical. Can be invoked from the command line.

Source Browsers

Version Control Integration Source Code

- ✧ Supports Microsoft Source Safe & others

Text Difference Engine

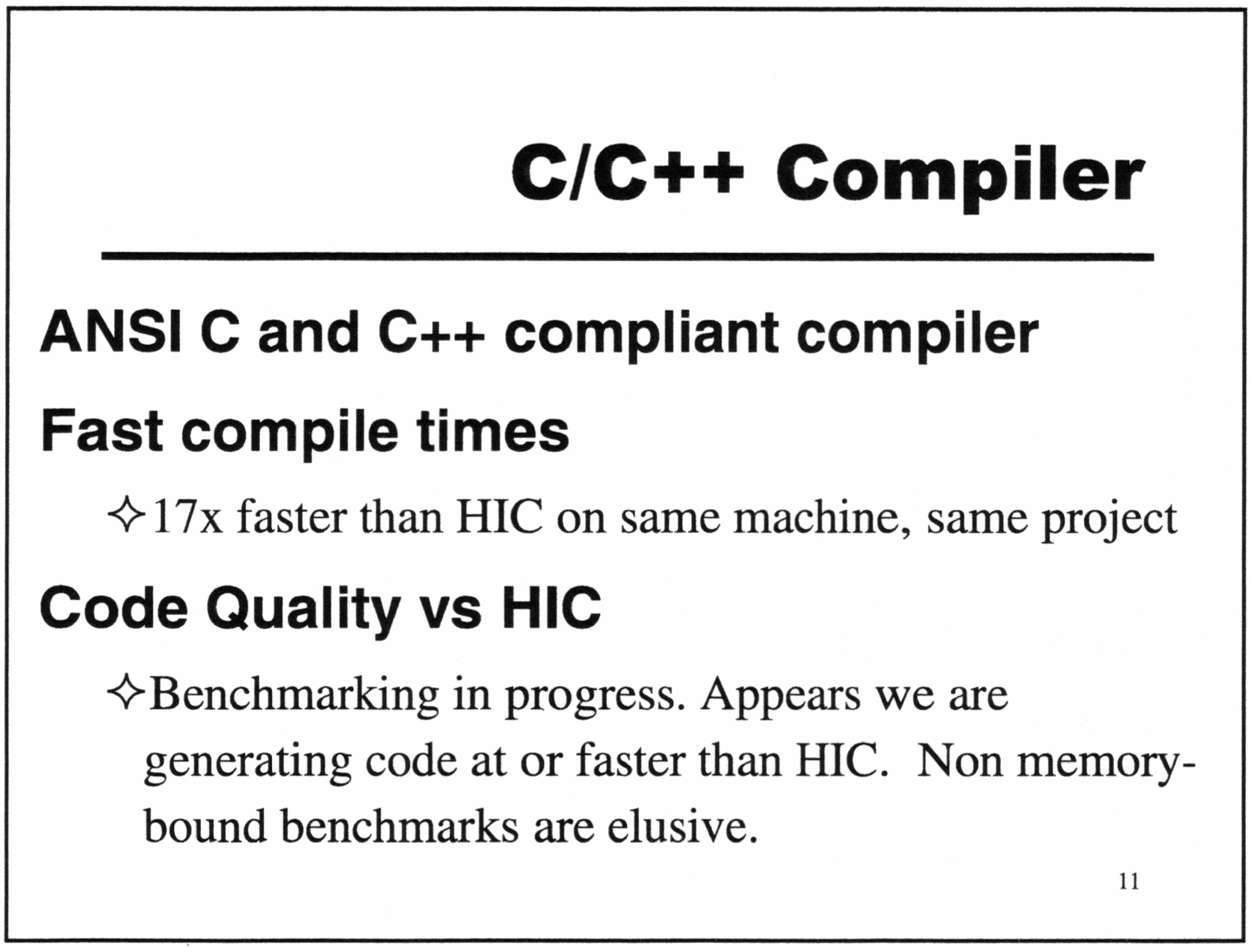


Metrowerks Build Tools

C/C++ Compiler

Assembler

Linker



C/C++ Compiler

ANSI C and C++ compliant compiler

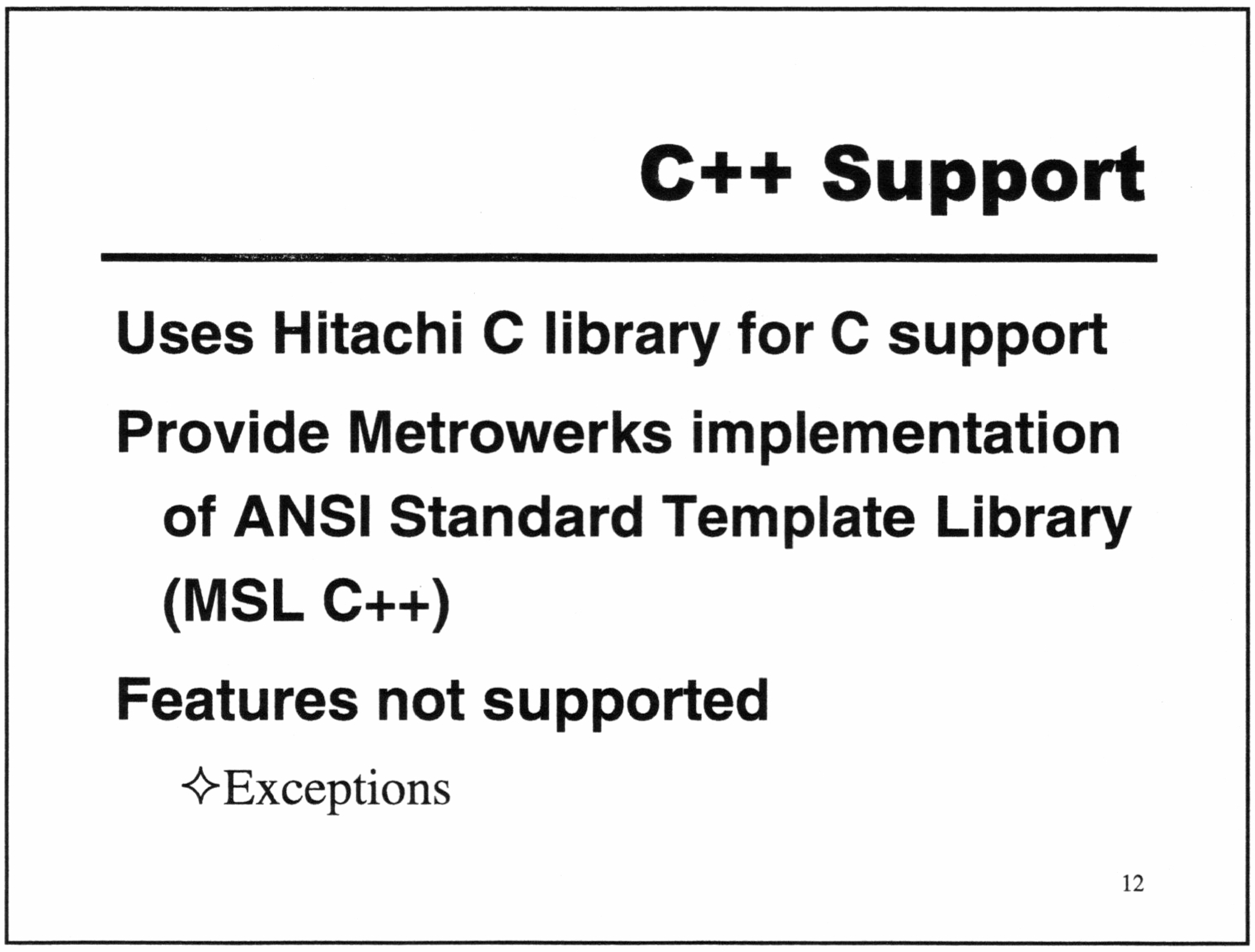
Fast compile times

- ✧17x faster than HIC on same machine, same project

Code Quality vs HIC

- ✧Benchmarking in progress. Appears we are generating code at or faster than HIC. Non memory-bound benchmarks are elusive.

11



C++ Support

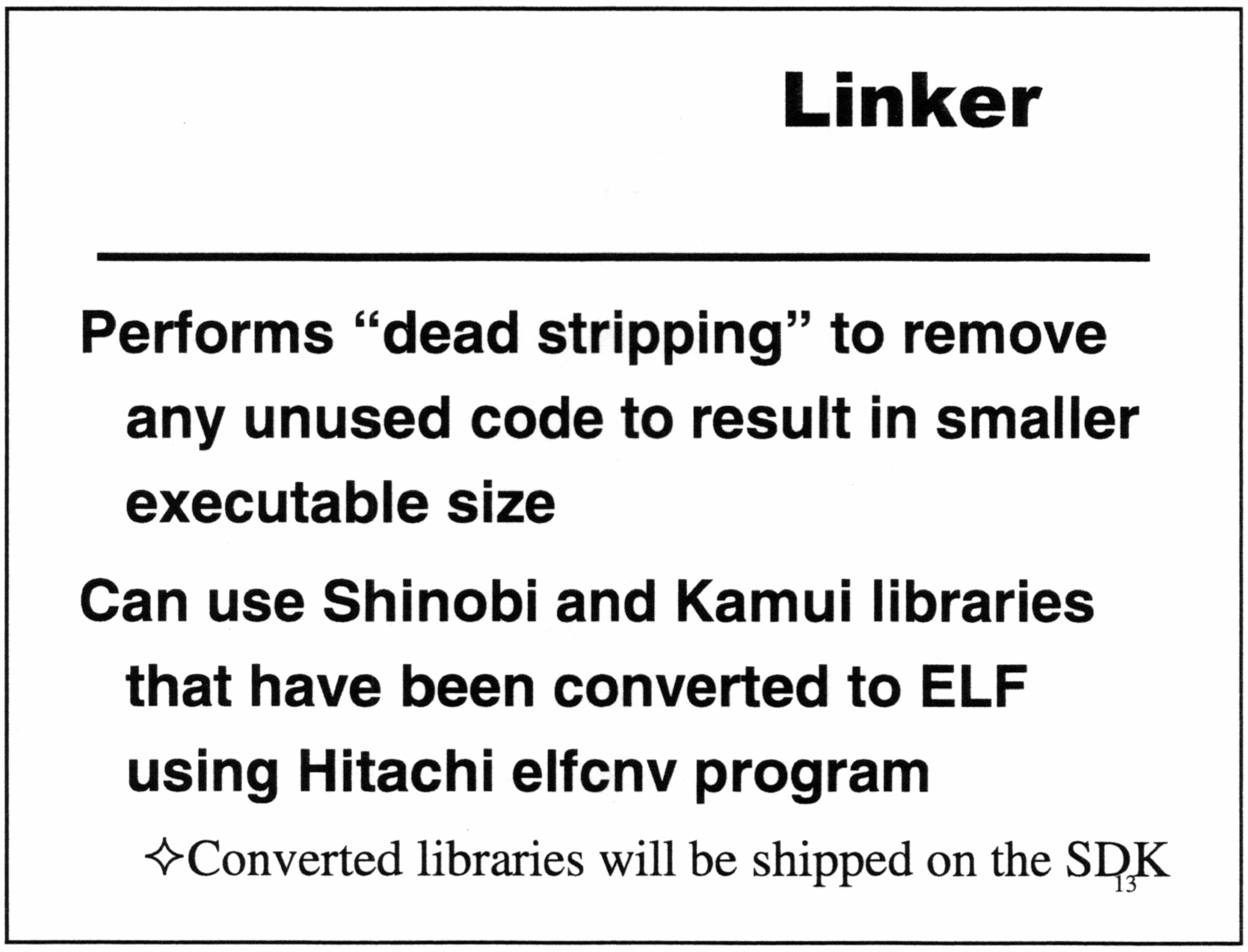
Uses Hitachi C library for C support

Provide Metrowerks implementation of ANSI Standard Template Library (MSL C++)

Features not supported

- ✧Exceptions

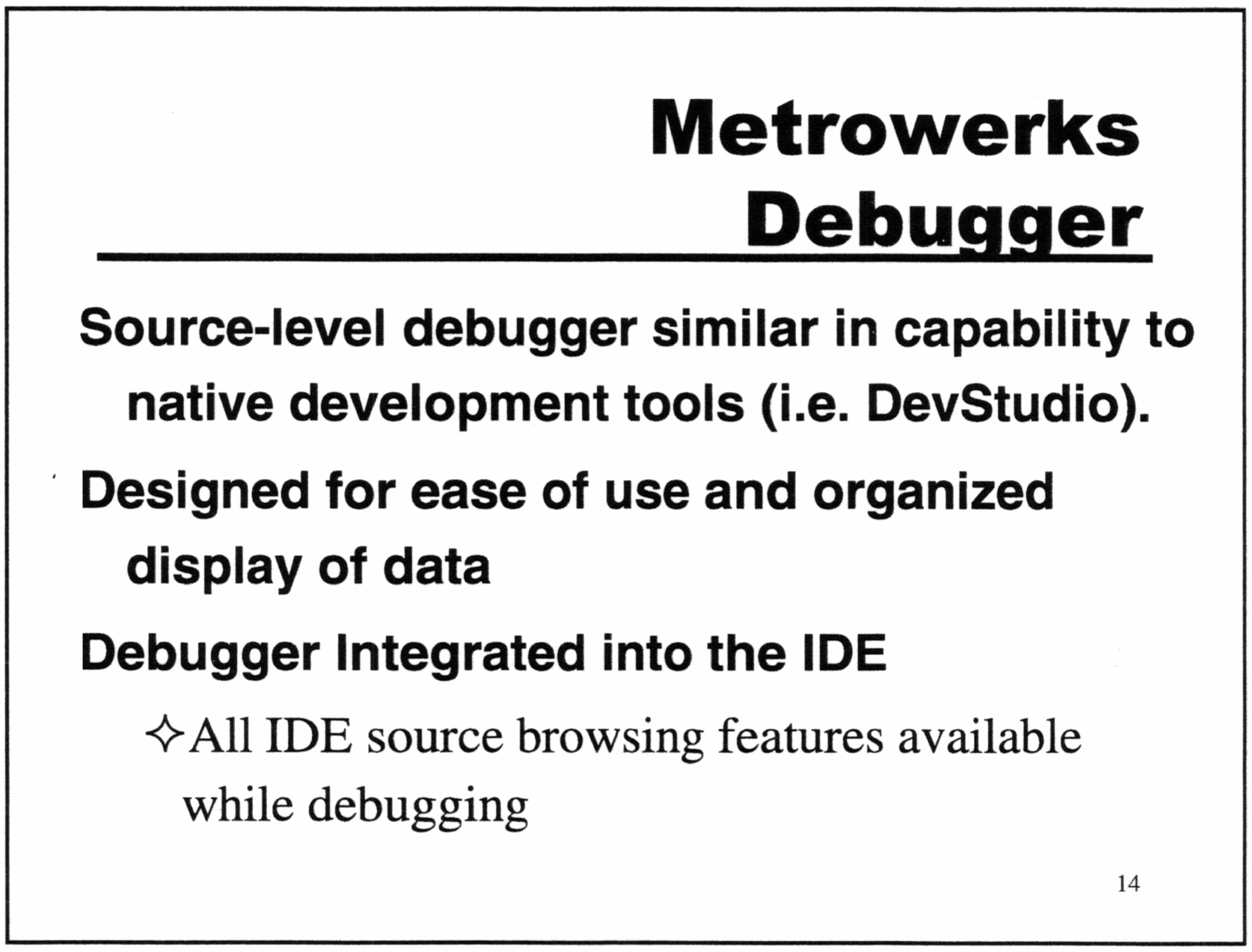
12



Linker

- Performs "dead stripping" to remove any unused code to result in smaller executable size
- Can use Shinobi and Kamui libraries that have been converted to ELF using Hitachi elfcnv program
  - ✧Converted libraries will be shipped on the SDK

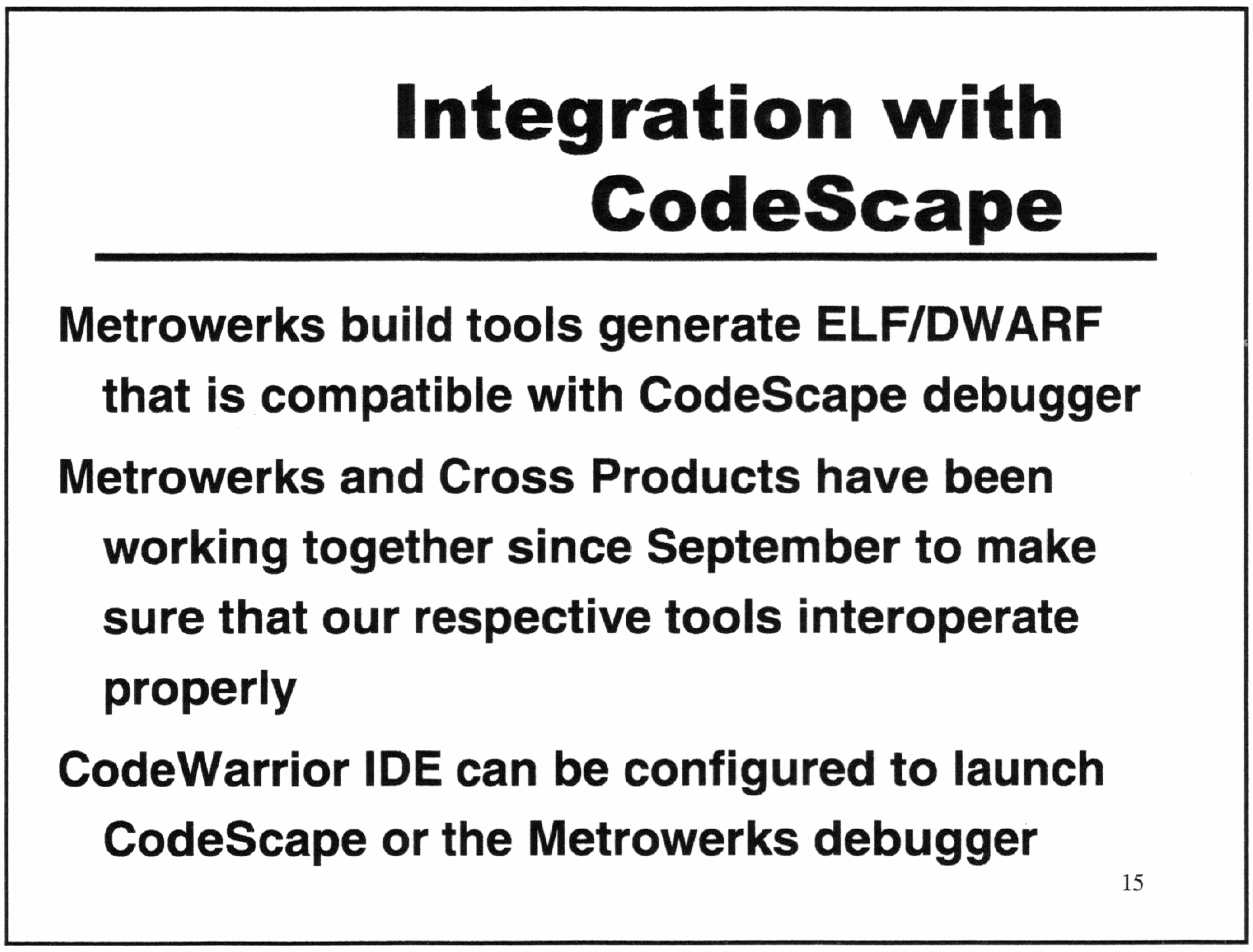
13



Metrowerks Debugger

- Source-level debugger similar in capability to native development tools (i.e. DevStudio).
- Designed for ease of use and organized display of data
- Debugger Integrated into the IDE
  - ✧All IDE source browsing features available while debugging

14

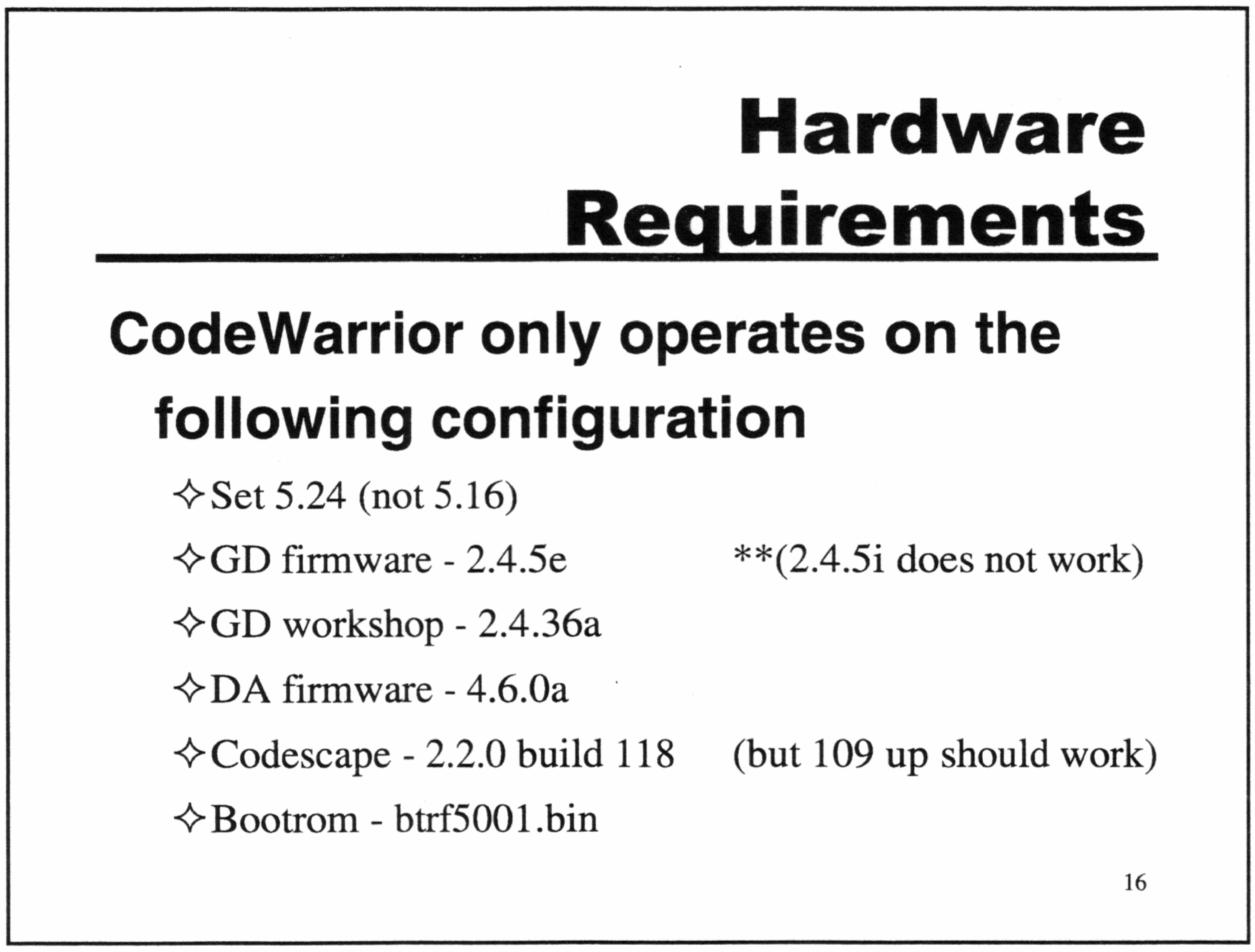


Integration with CodeScape

Metrowerks build tools generate ELF/DWARF that is compatible with CodeScape debugger

Metrowerks and Cross Products have been working together since September to make sure that our respective tools interoperate properly

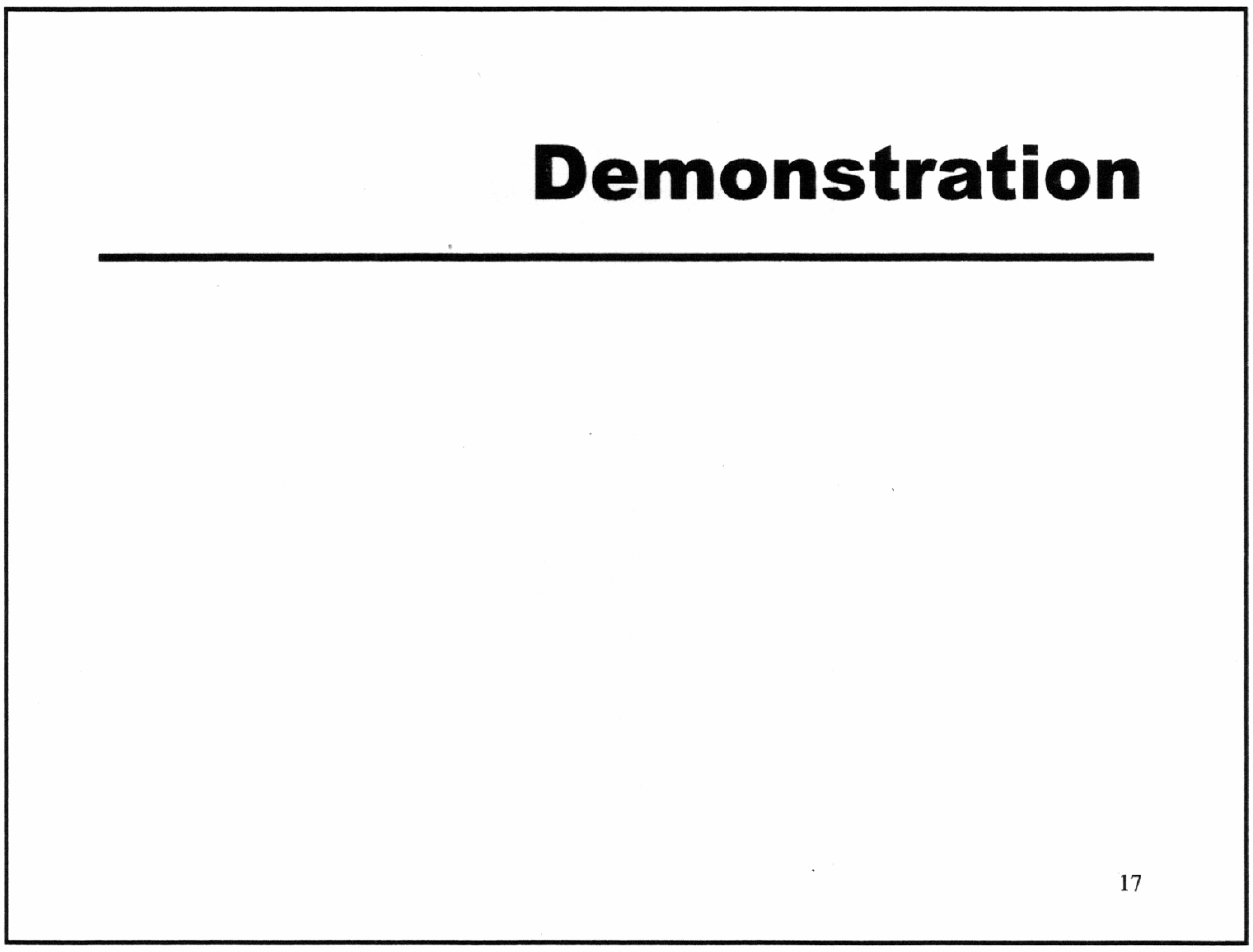
CodeWarrior IDE can be configured to launch CodeScape or the Metrowerks debugger



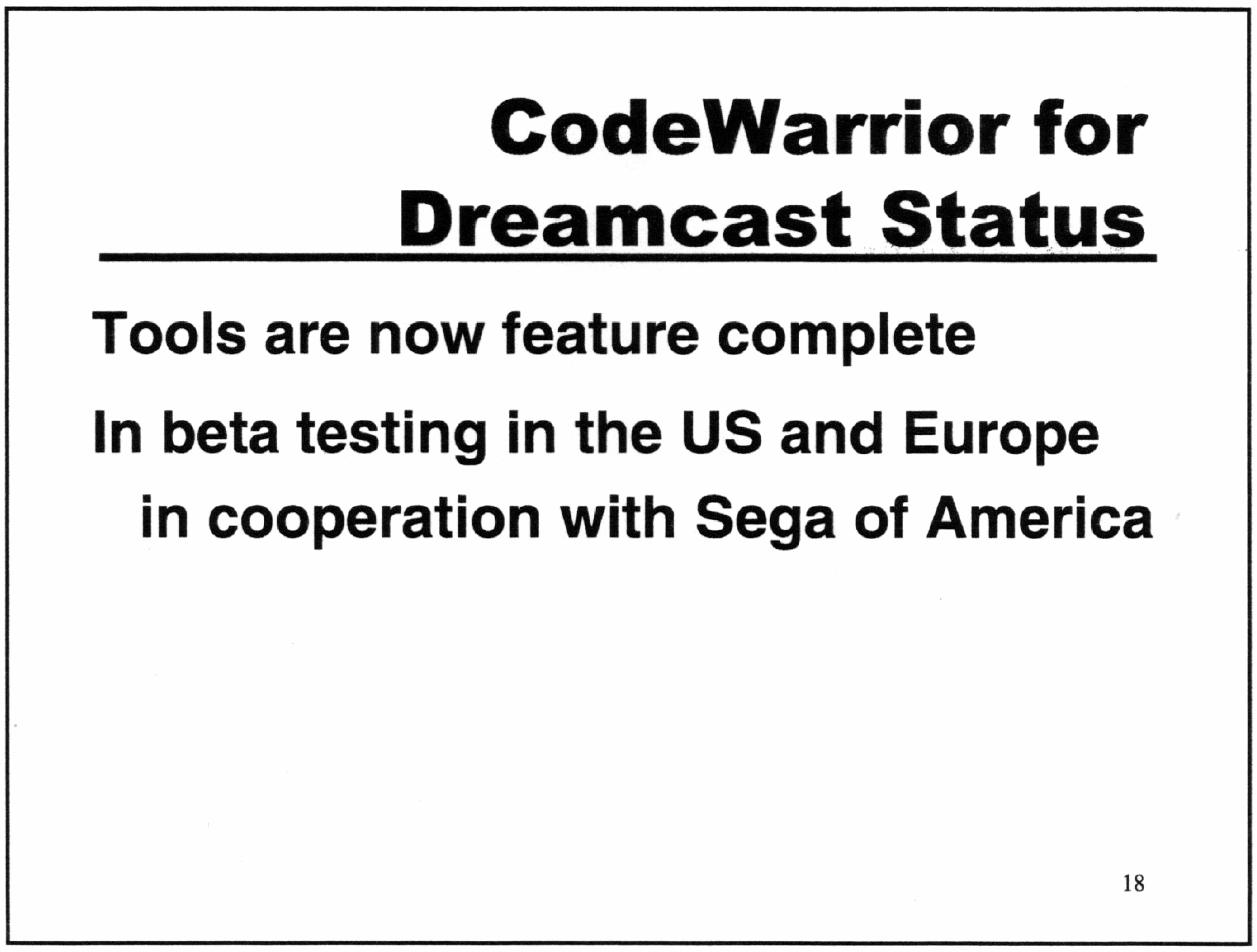
Hardware Requirements

CodeWarrior only operates on the following configuration

✧Set 5.24 (not 5.16)
✧GD firmware - 2.4.5e **(2.4.5i does not work)
✧GD workshop - 2.4.36a
✧DA firmware - 4.6.0a
✧Codescape - 2.2.0 build 118 (but 109 up should work)
✧Bootrom - btrf5001.bin



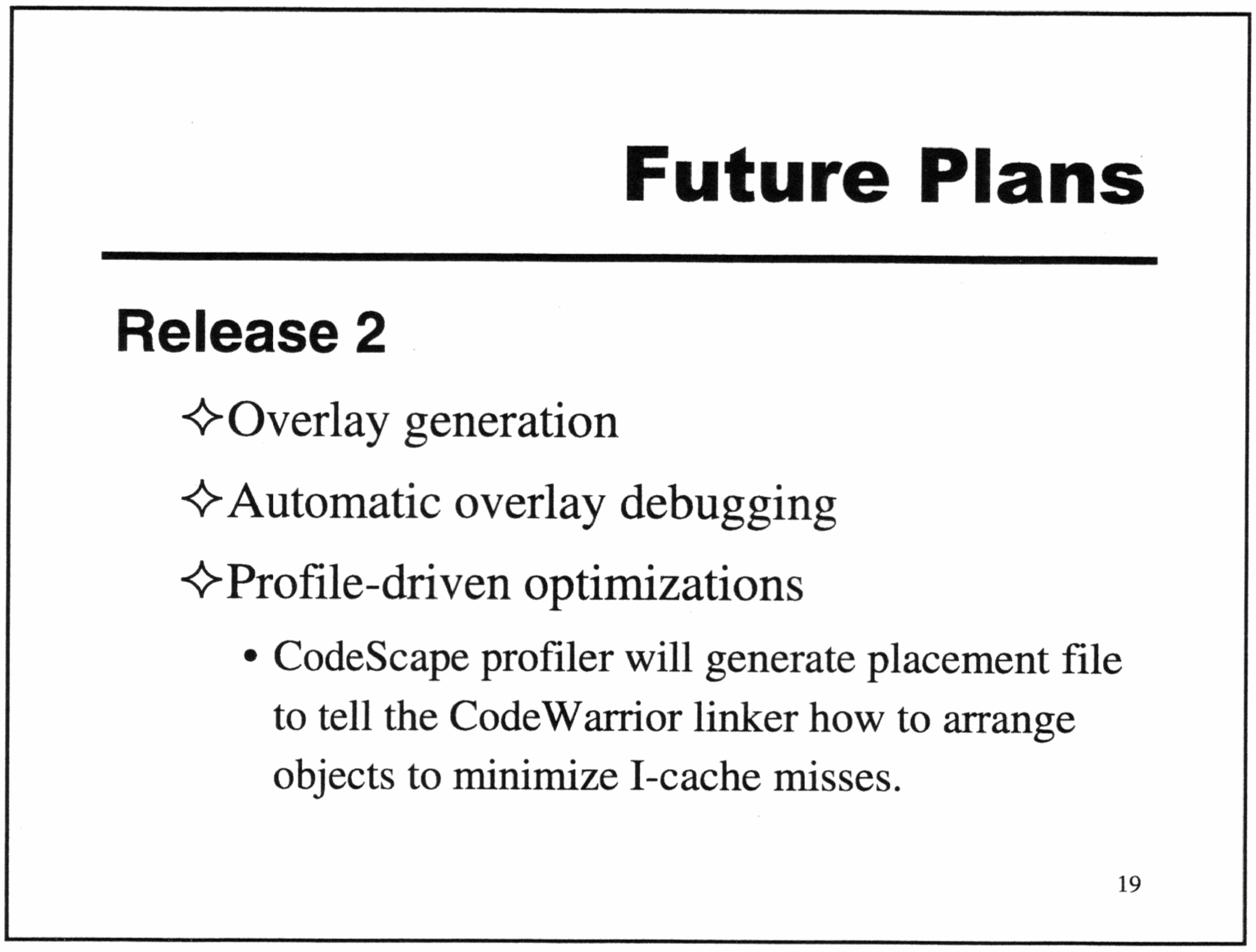
Demonstration



CodeWarrior for Dreamcast Status

Tools are now feature complete

In beta testing in the US and Europe
in cooperation with Sega of America



Future Plans

Release 2

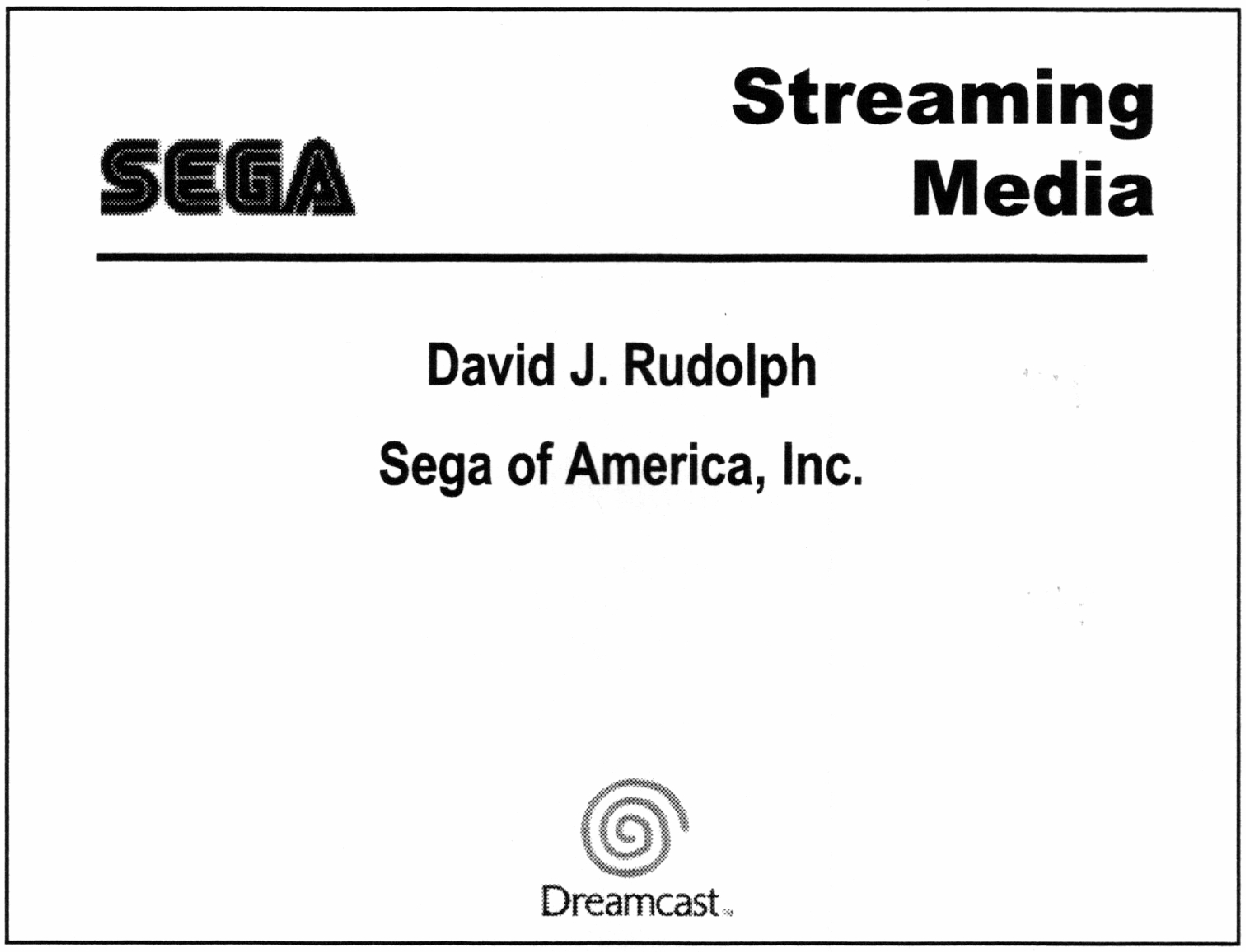
- ✧Overlay generation
- ✧Automatic overlay debugging
- ✧Profile-driven optimizations
  - CodeScape profiler will generate placement file to tell the CodeWarrior linker how to arrange objects to minimize I-cache misses.

SEGA

Streaming Media

David J. Rudolph
Sega of America

Dreamcast™



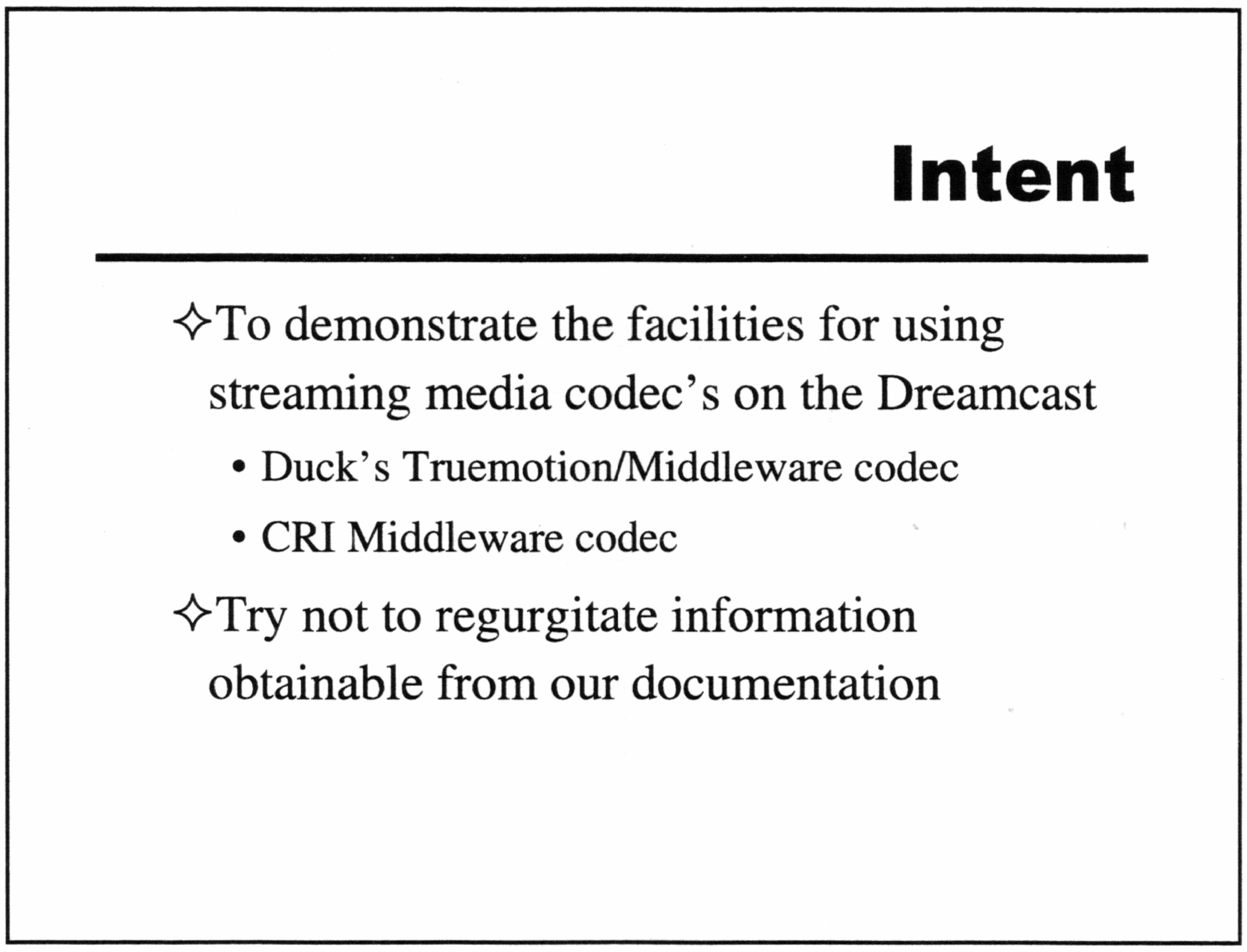
SEGA

Streaming Media

David J. Rudolph

Sega of America, Inc.

Dreamcast

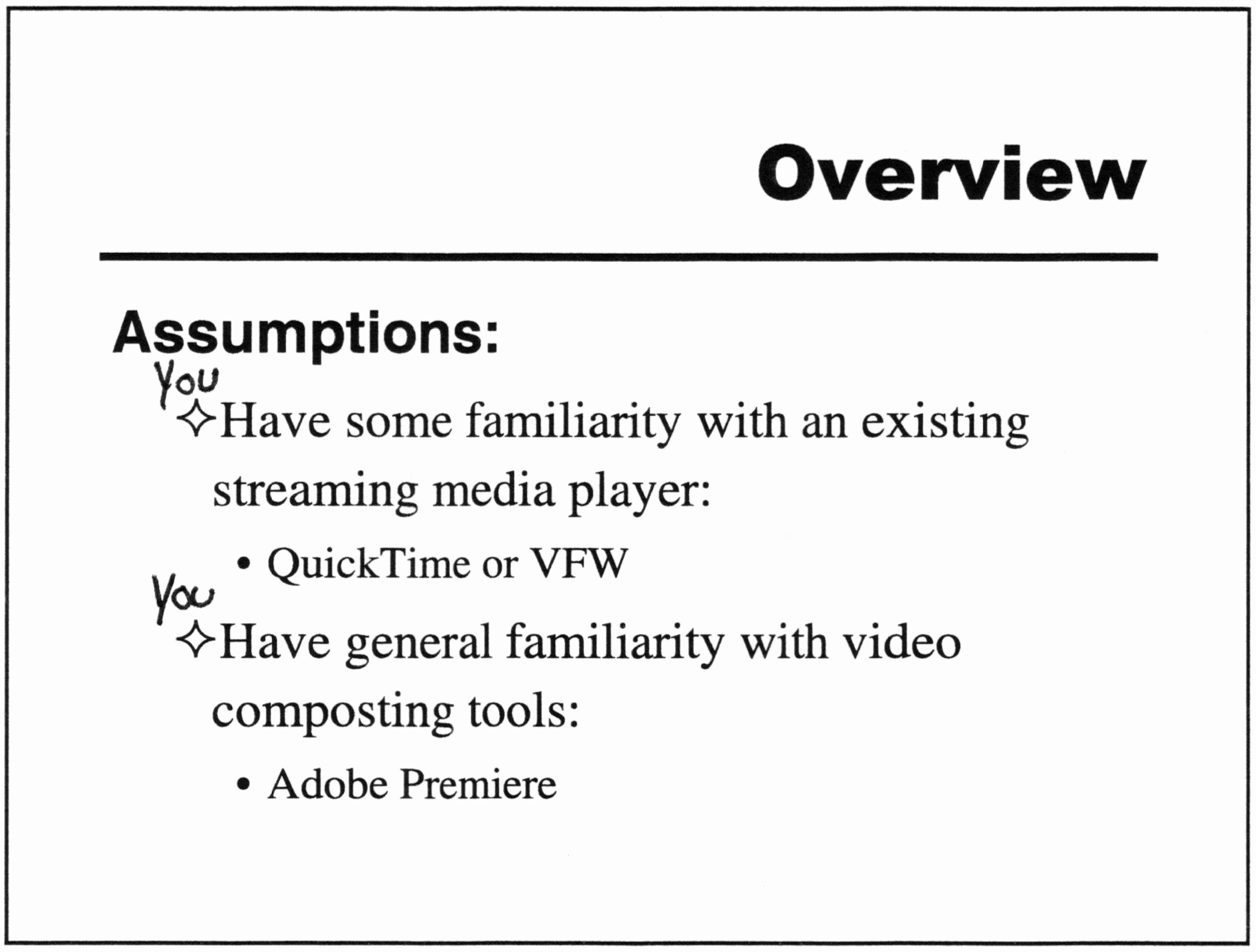


Intent

✧To demonstrate the facilities for using streaming media codec's on the Dreamcast

- Duck's Truemotion/Middleware codec
- CRI Middleware codec

✧Try not to regurgitate information obtainable from our documentation



Overview

Assumptions:

You

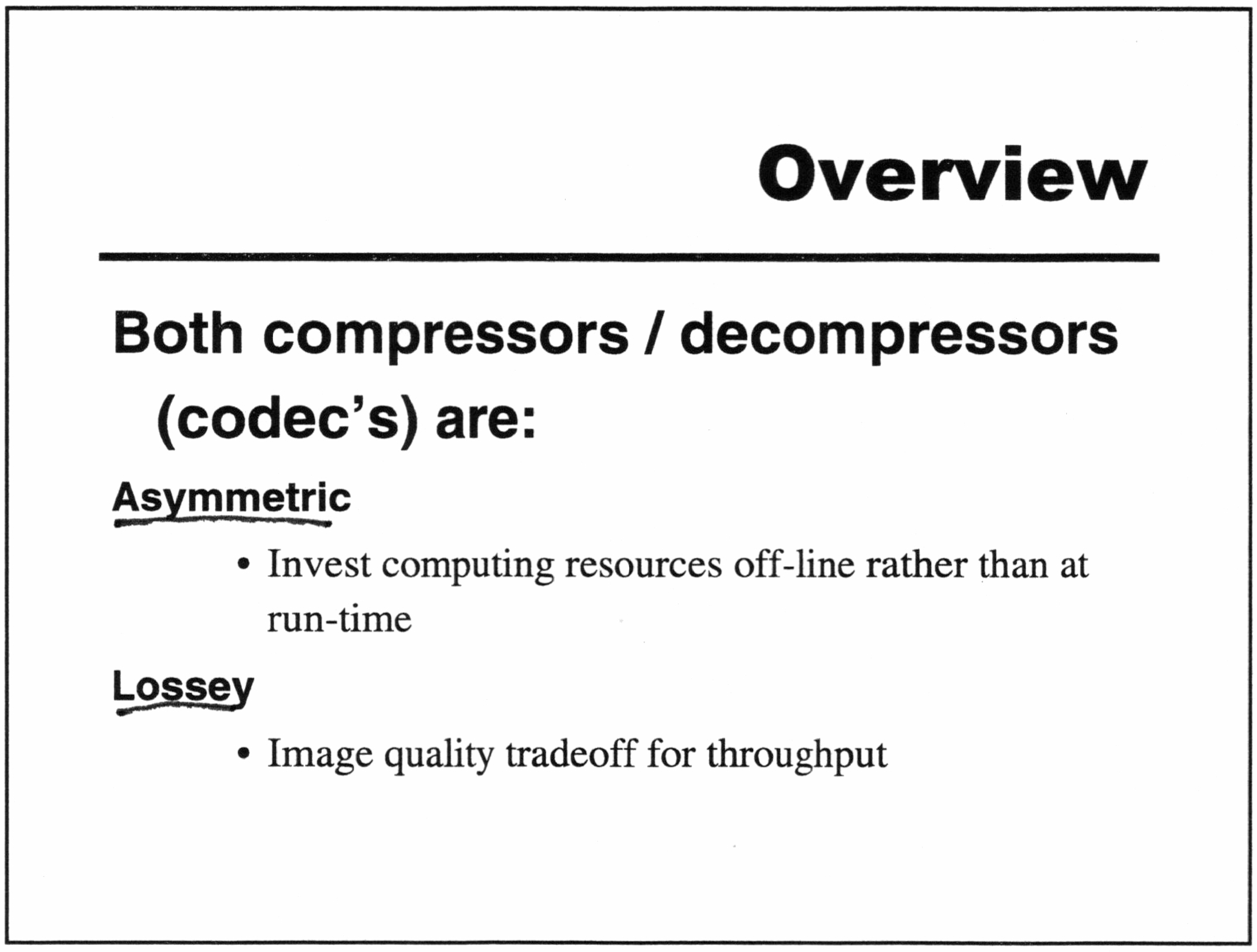
✧Have some familiarity with an existing streaming media player:

- QuickTime or VFW

You

✧Have general familiarity with video composting tools:

- Adobe Premiere



Overview

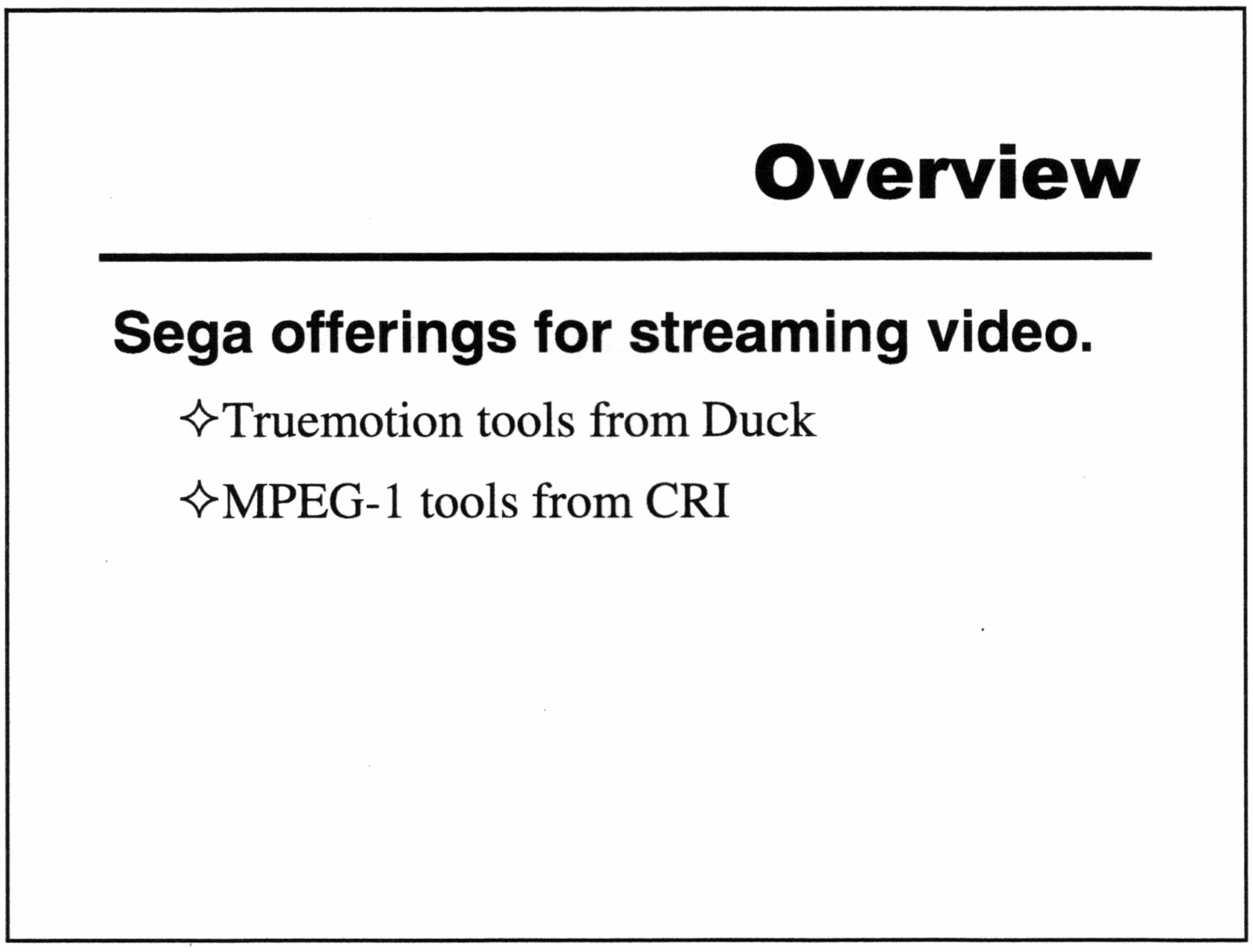
Both compressors / decompressors (codec's) are:

Asymmetric

- Invest computing resources off-line rather than at run-time

Lossey

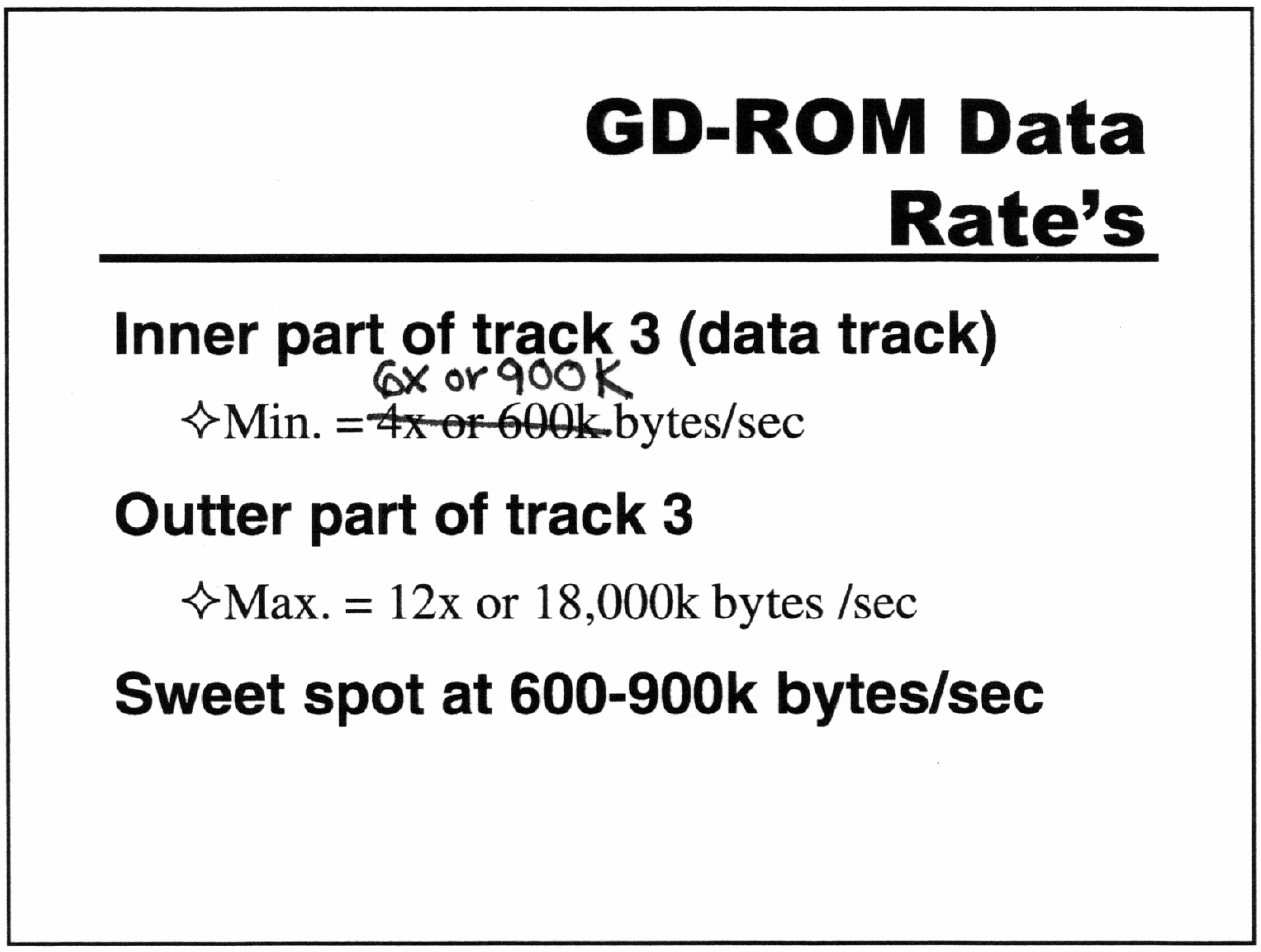
- Image quality tradeoff for throughput



Overview

Sega offerings for streaming video.

- ✧Truemotion tools from Duck
- ✧MPEG-1 tools from CRI



GD-ROM Data Rate’s

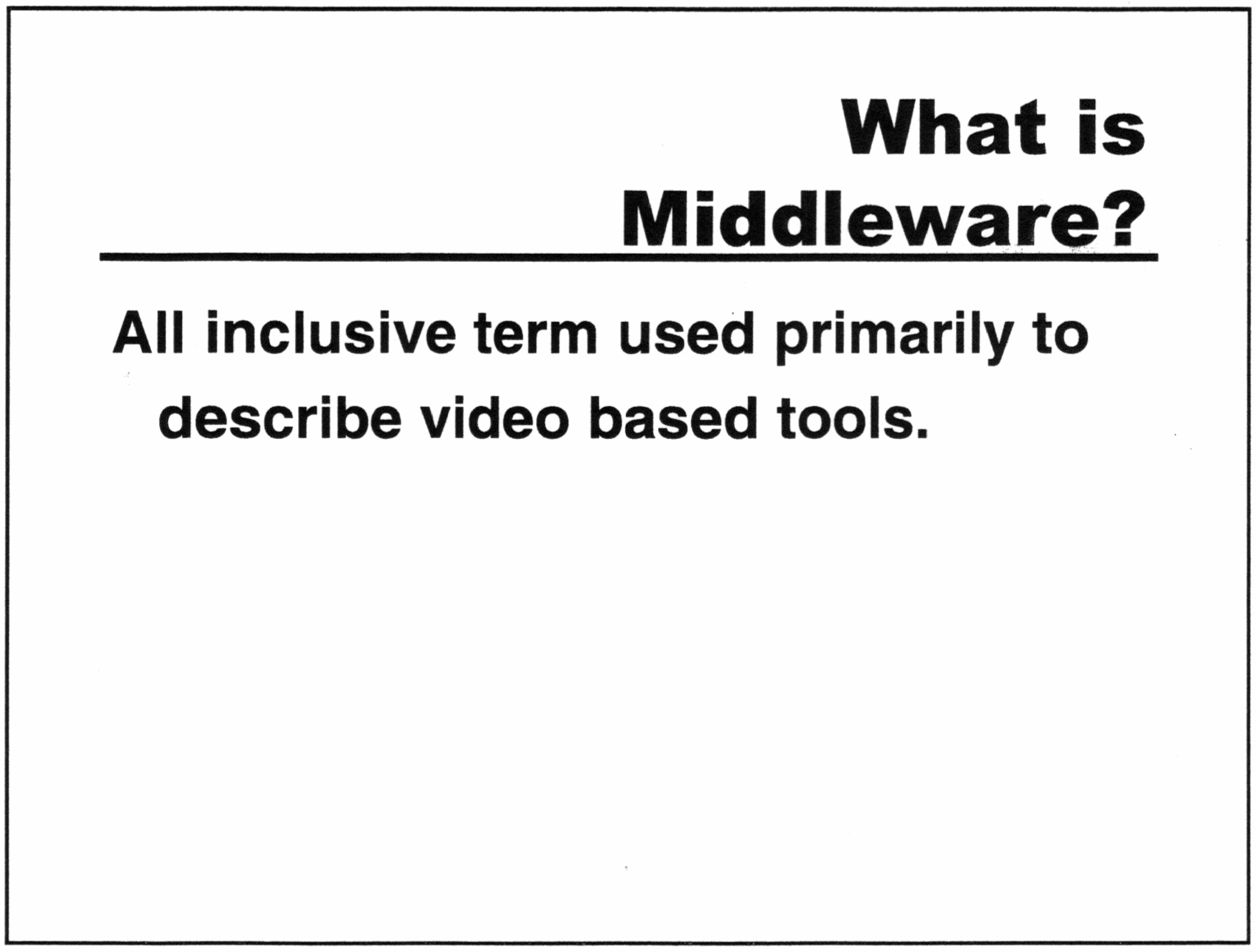
Inner part of track 3 (data track)

- ✧Min. = ~~4x or 600k~~ 6x or 900K bytes/sec

Outter part of track 3

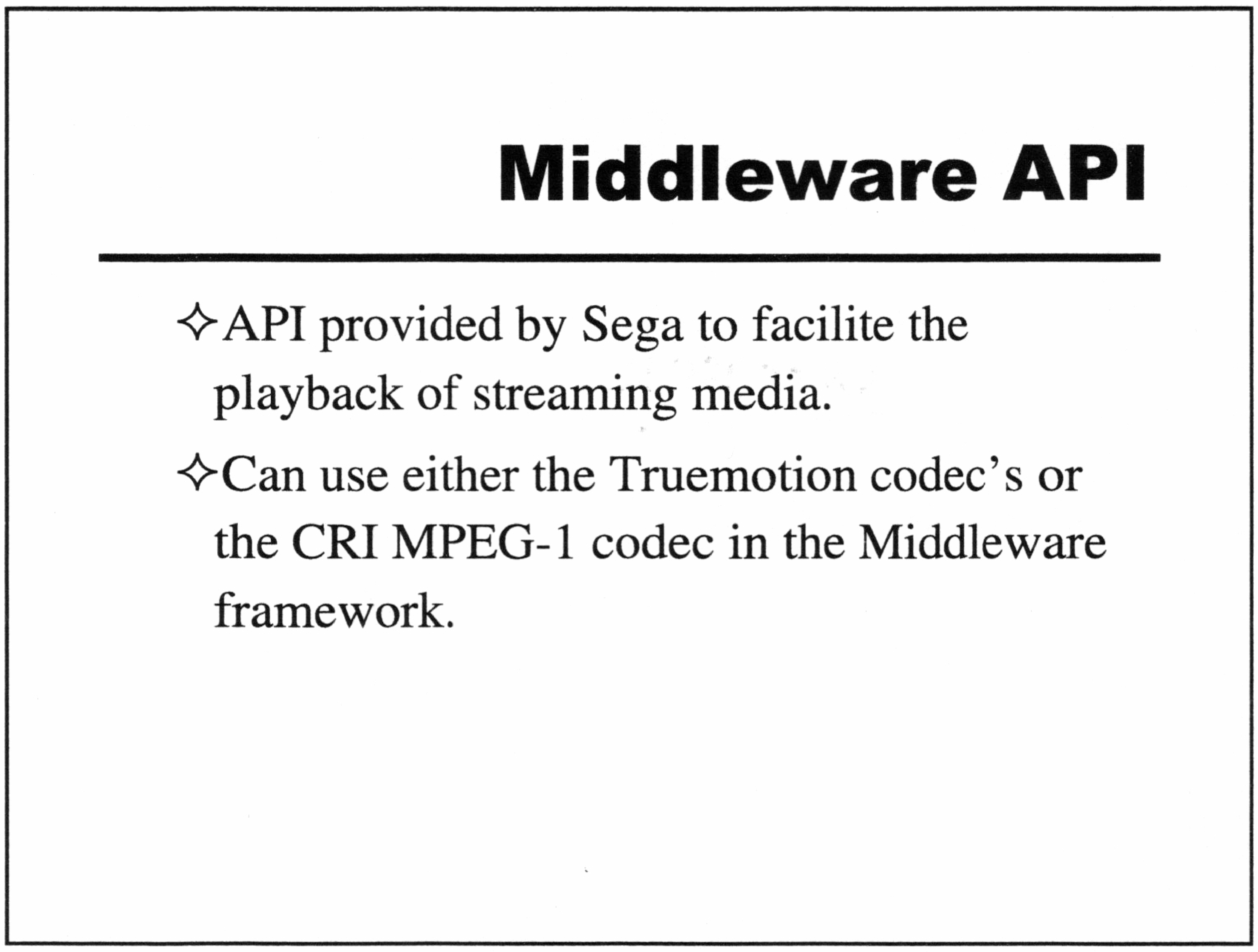
- ✧Max. = 12x or 18,000k bytes /sec

Sweet spot at 600-900k bytes/sec



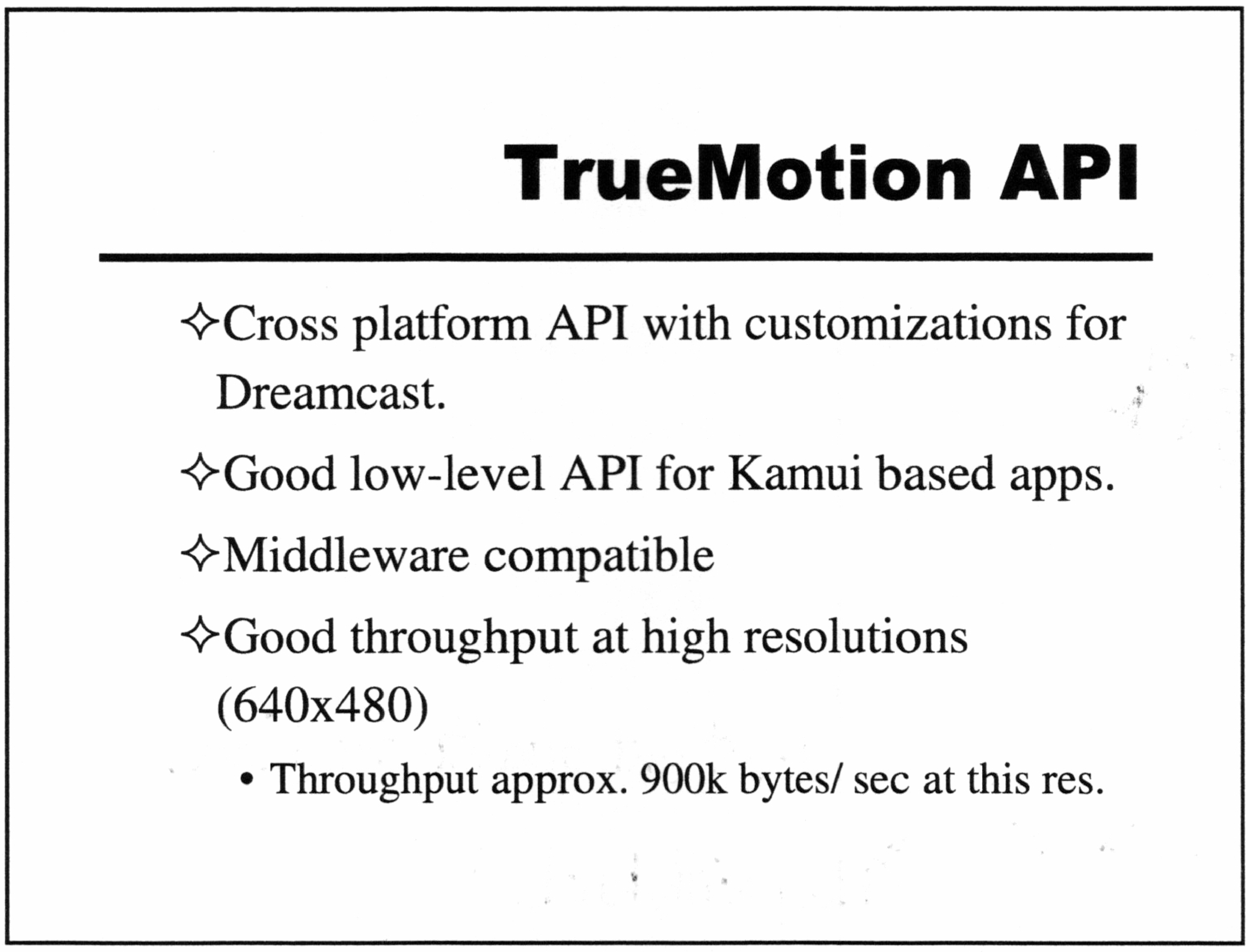
What is Middleware?

All inclusive term used primarily to describe video based tools.



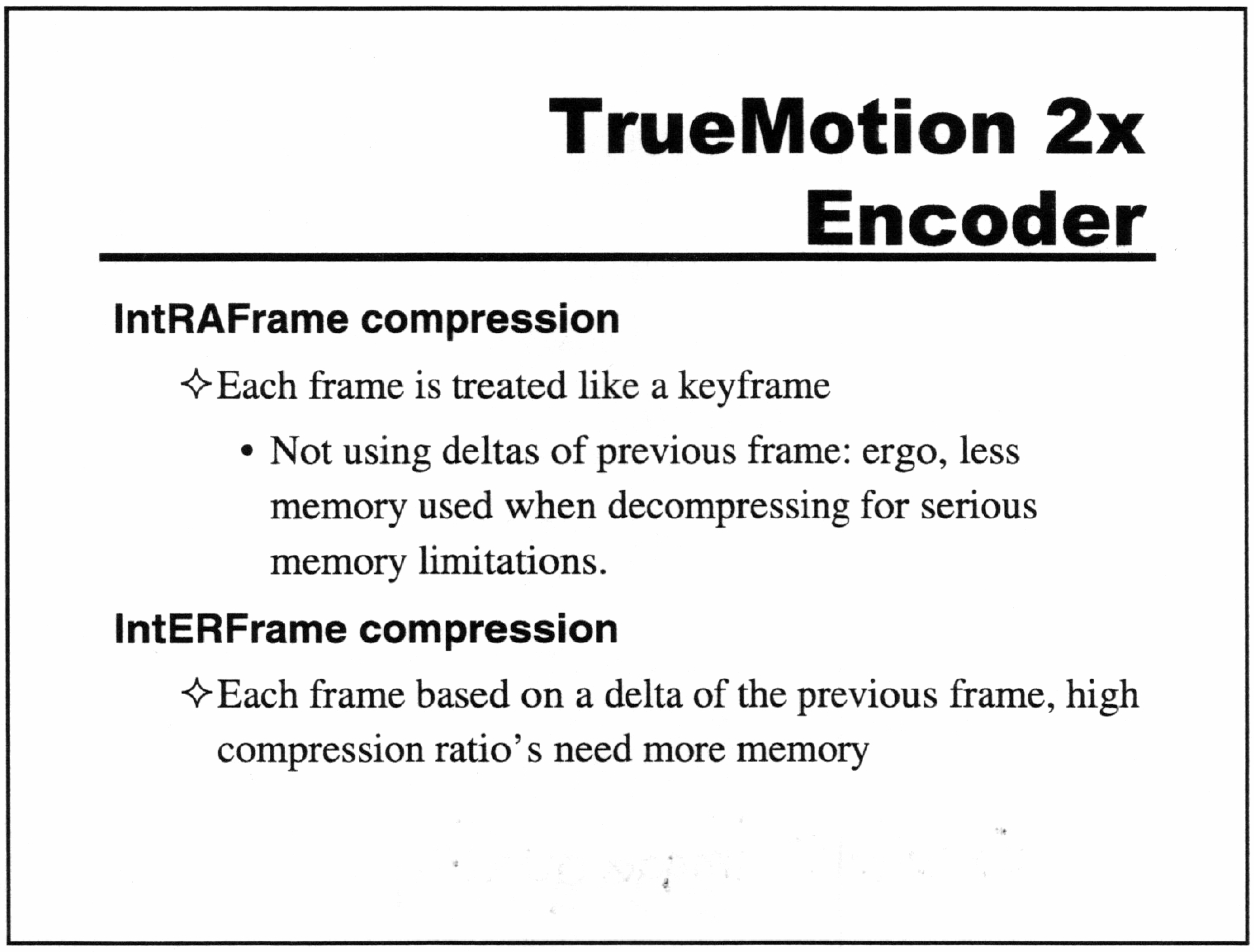
Middleware API

- ✧API provided by Sega to facilite the playback of streaming media.
- ✧Can use either the Truemotion codec's or the CRI MPEG-1 codec in the Middleware framework.



TrueMotion API

- ✧Cross platform API with customizations for Dreamcast.
- ✧Good low-level API for Kamui based apps.
- ✧Middleware compatible
- ✧Good throughput at high resolutions (640x480)
  - Throughput approx. 900k bytes/ sec at this res.



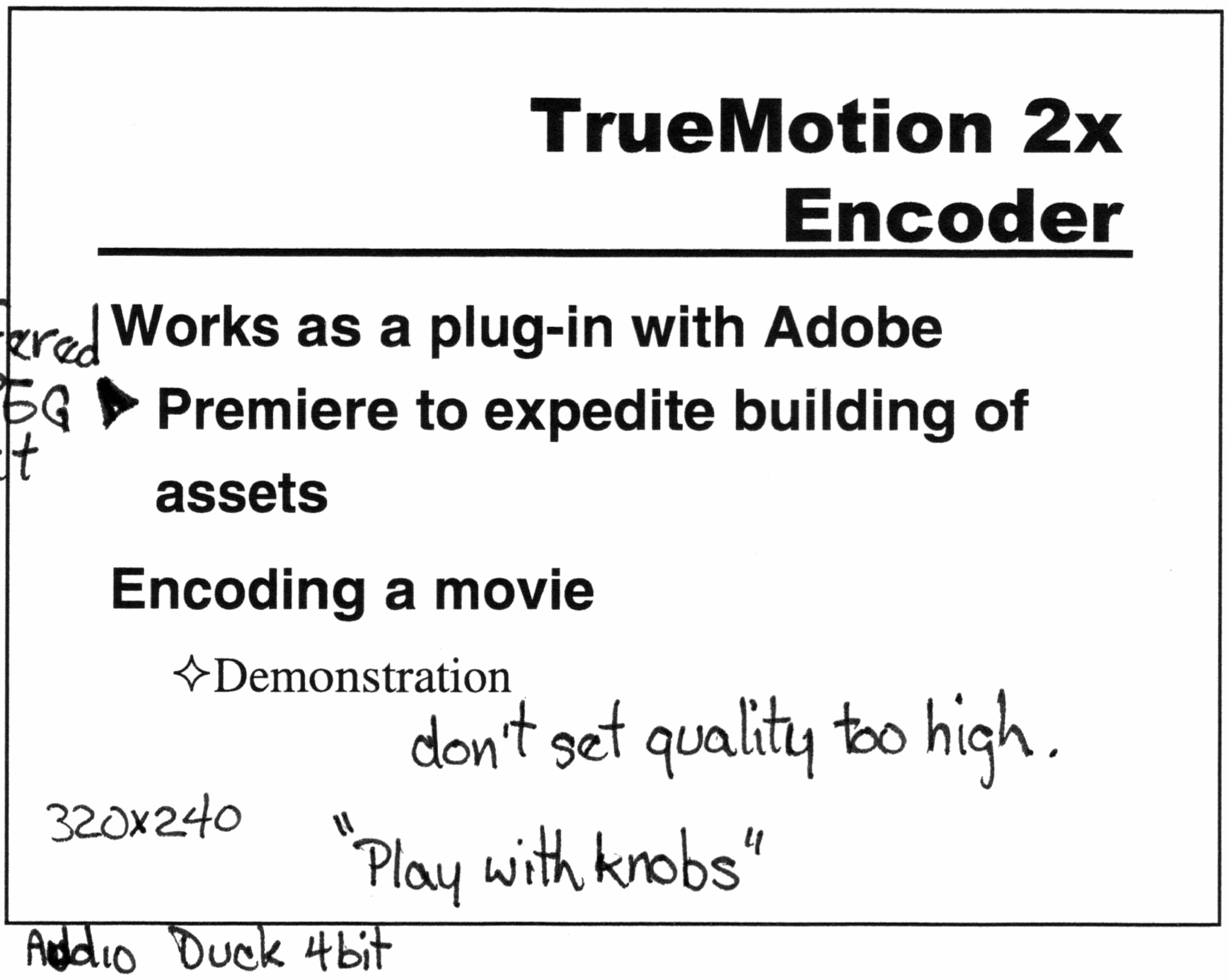
TrueMotion 2x Encoder

IntRAFrame compression

- ✧Each frame is treated like a keyframe
  - Not using deltas of previous frame: ergo, less memory used when decompressing for serious memory limitations.

IntERFrame compression

- ✧Each frame based on a delta of the previous frame, high compression ratio's need more memory



TrueMotion 2x Encoder

~~Do~~ · Not offered for MPEG yet

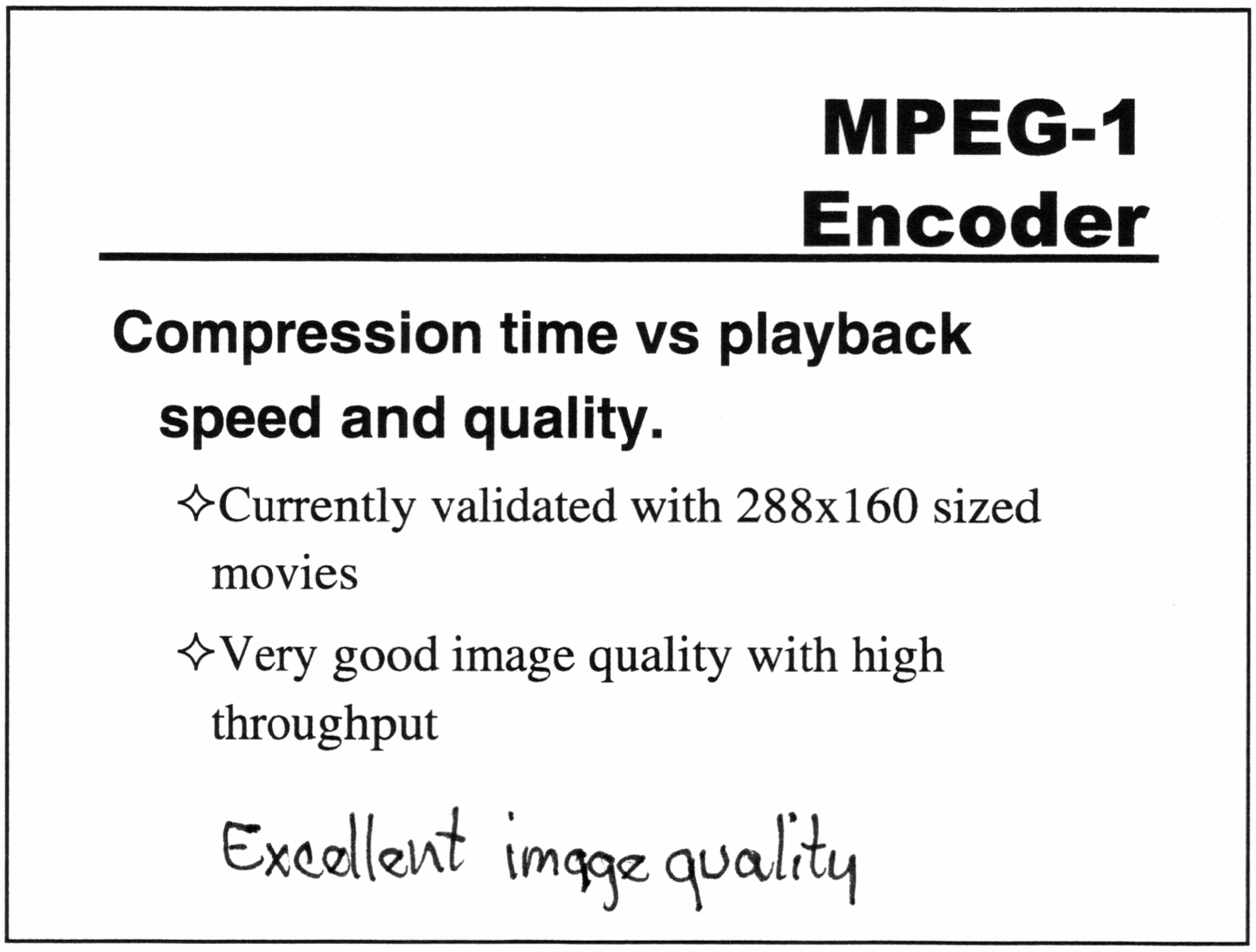
- Works as a plug-in with Adobe Premiere to expedite building of assets
- Encoding a movie
  - ✧Demonstration

don't set quality too high.

320x240

"Play with knobs"

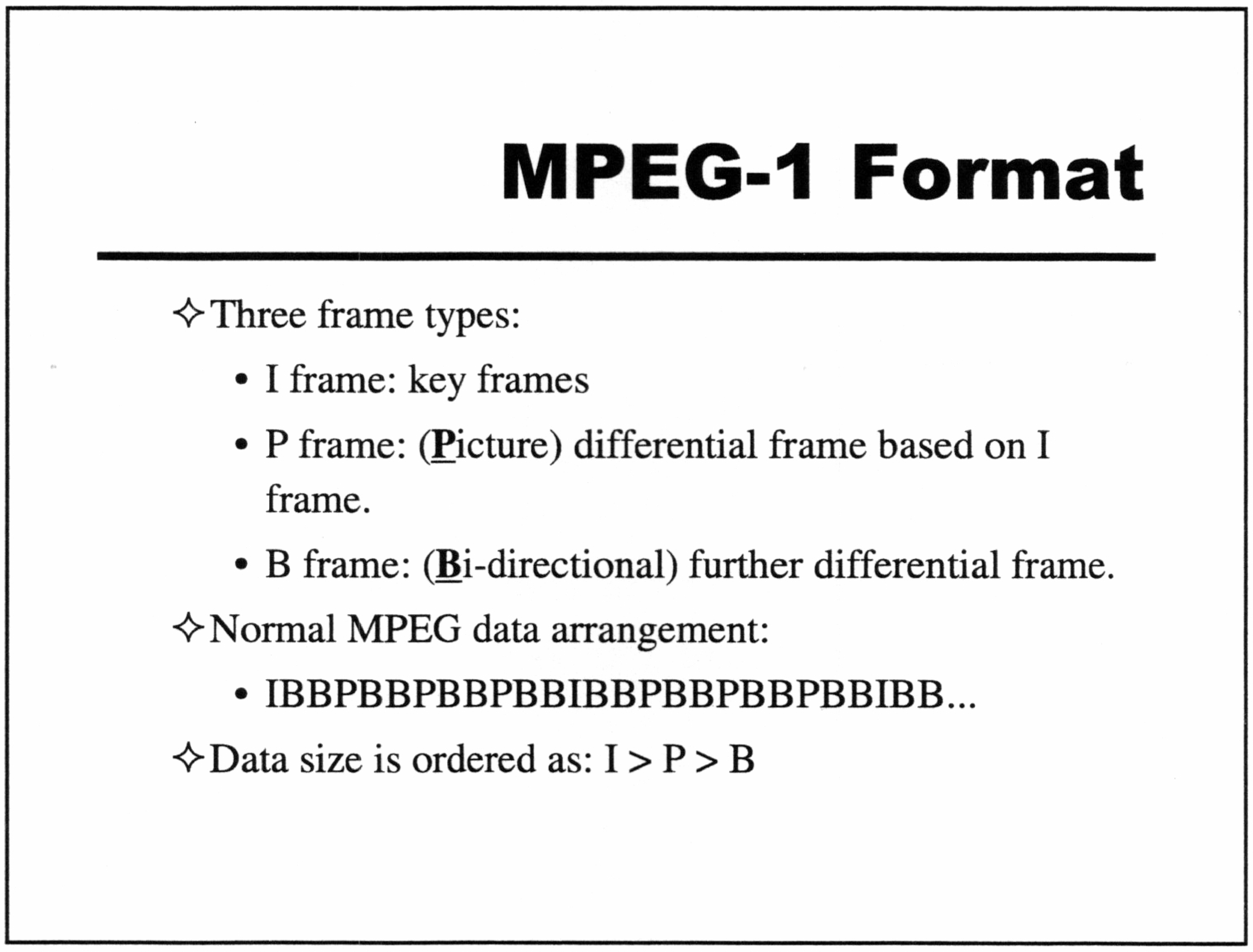
Audio Duck 4bit



MPEG-1 Encoder

- Compression time vs playback speed and quality.
  - ✧Currently validated with 288x160 sized movies
  - ✧Very good image quality with high throughput

Excellent image quality

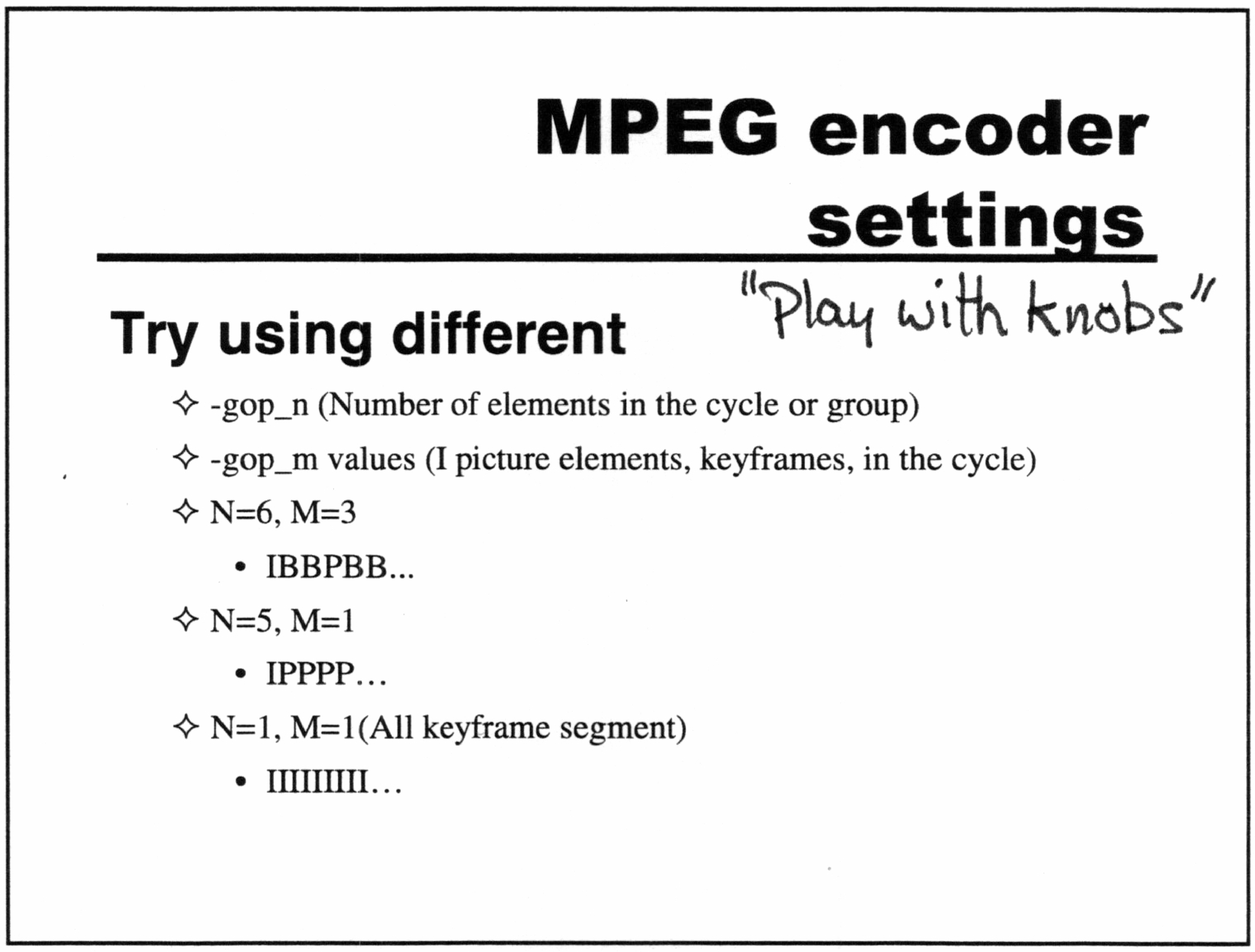


MPEG-1 Format

✧ Three frame types:
  - I frame: key frames
  - P frame: (**P**icture) differential frame based on I frame.
  - B frame: (**B**i-directional) further differential frame.

✧ Normal MPEG data arrangement:
  - IBBPBBPBBPBBIBBPBBPBBPBBIBB...

✧ Data size is ordered as: I > P > B



MPEG encoder settings

Try using different

"Play with knobs"

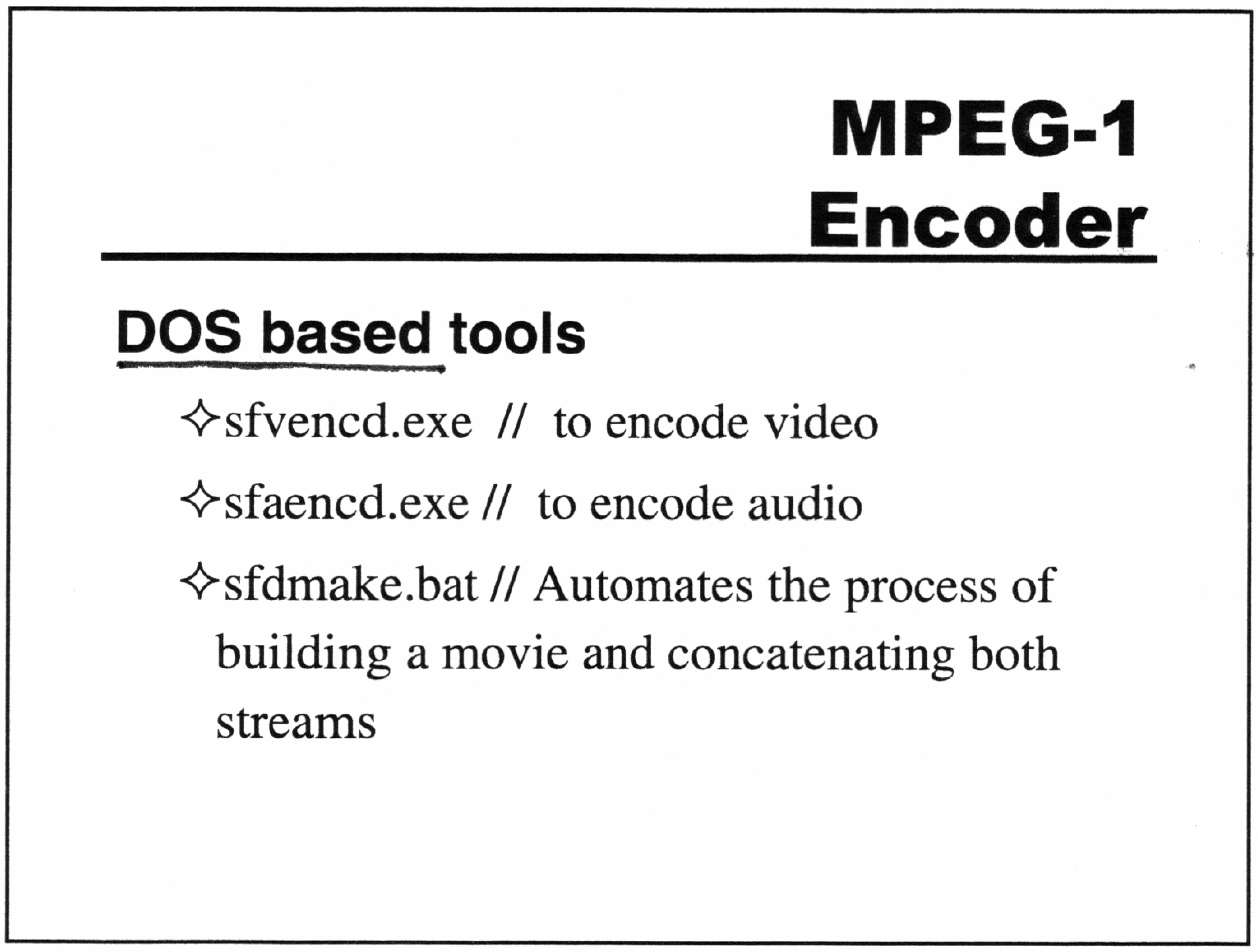
✧ -gop_n (Number of elements in the cycle or group)

✧ -gop_m values (I picture elements, keyframes, in the cycle)

✧ N=6, M=3
  - IBBPBB...

✧ N=5, M=1
  - IPPPP…

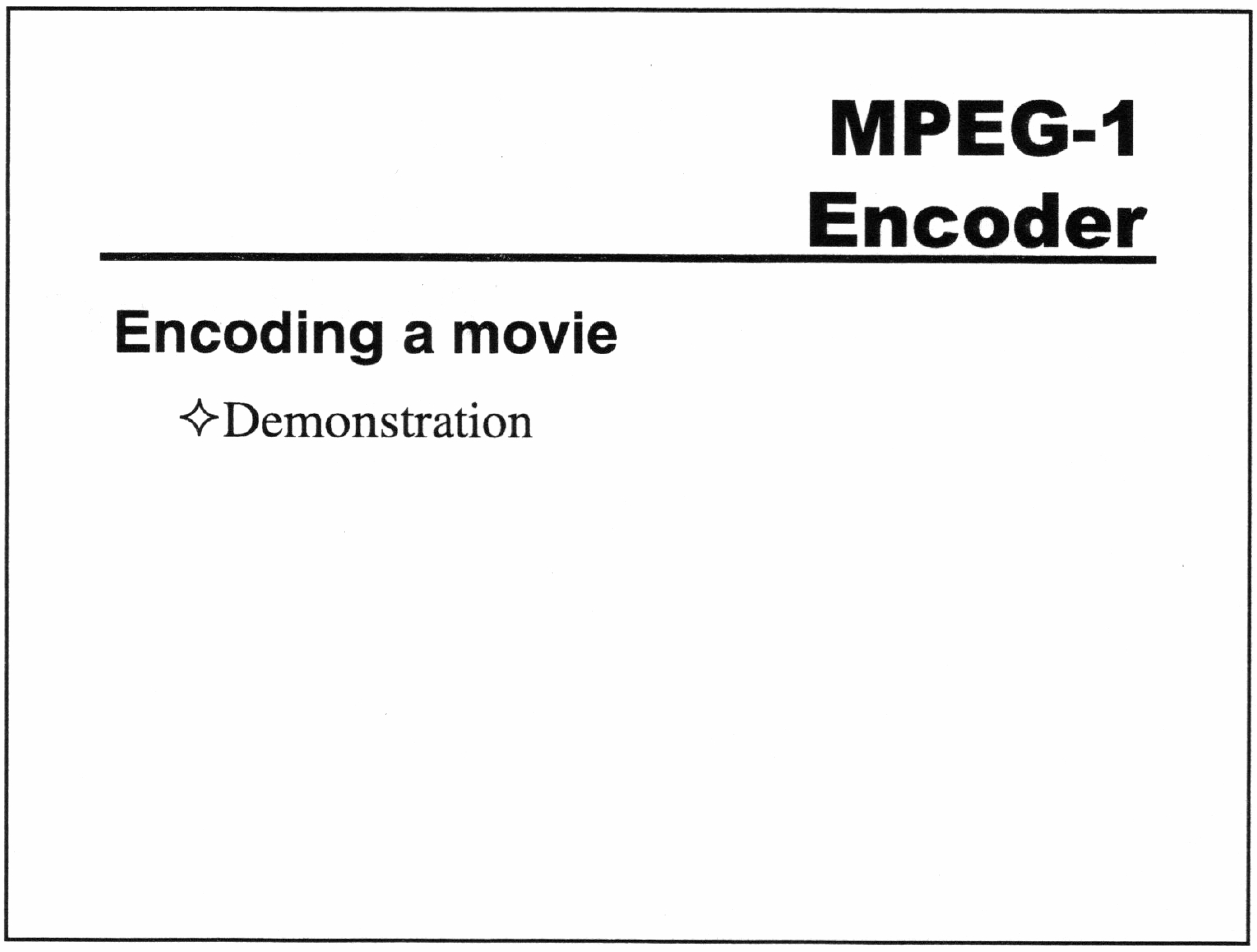
✧ N=1, M=1(All keyframe segment)
  - IIIIIIII…



MPEG-1 Encoder

DOS based tools

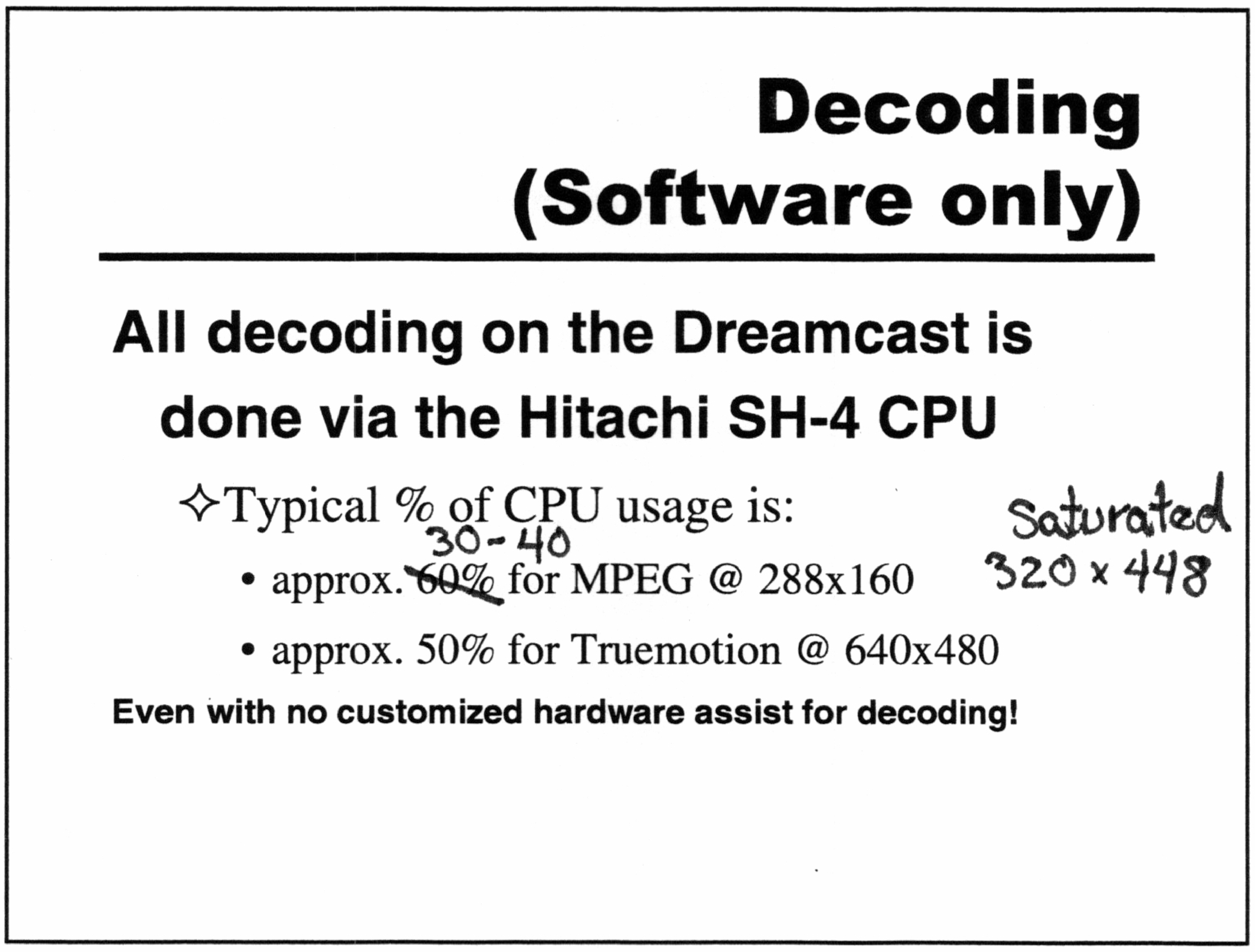
- ✧sfvencd.exe // to encode video
- ✧sfaencd.exe // to encode audio
- ✧sfdmake.bat // Automates the process of building a movie and concatenating both streams



MPEG-1 Encoder

Encoding a movie

- ✧Demonstration



Decoding (Software only)

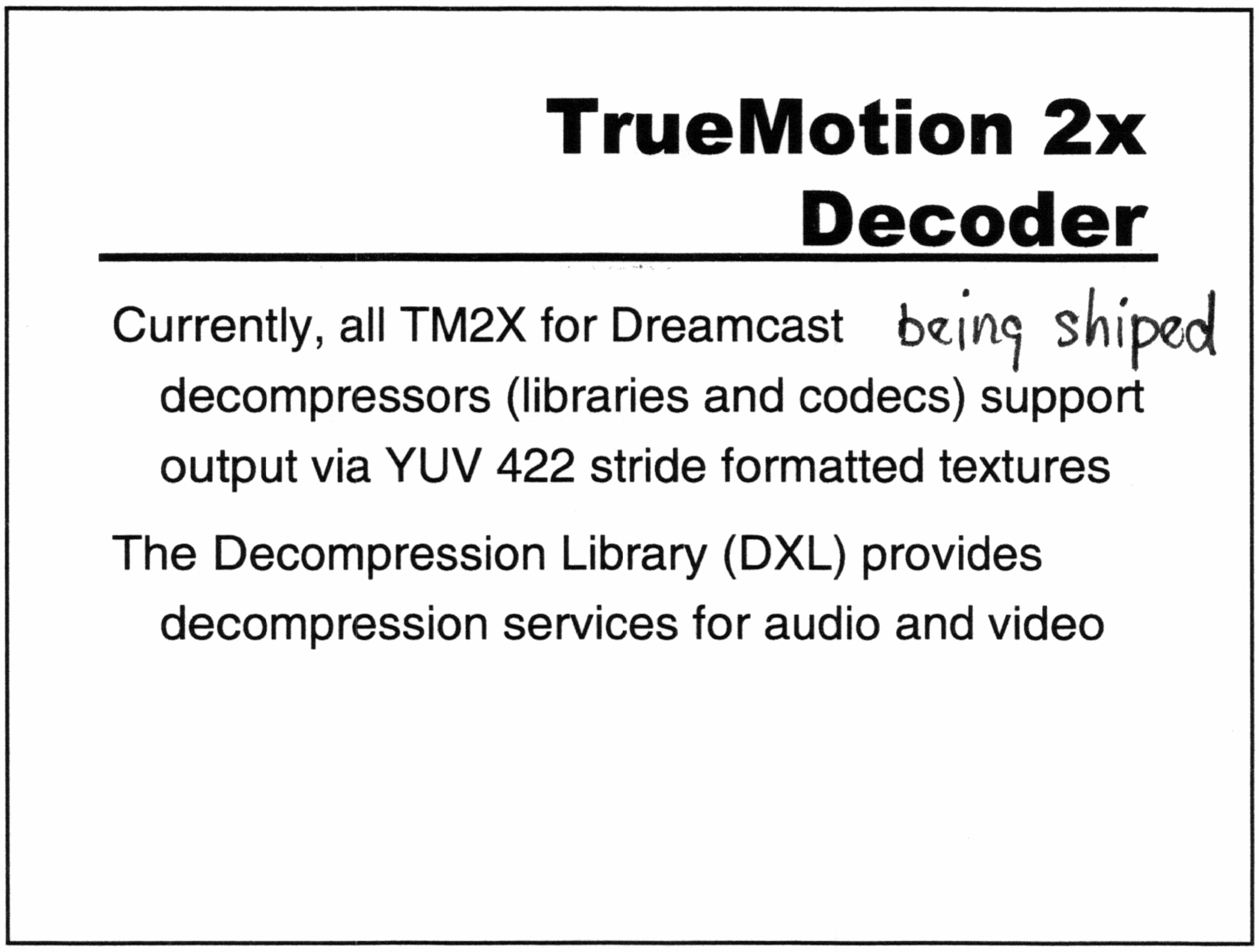
All decoding on the Dreamcast is done via the Hitachi SH-4 CPU

✧Typical % of CPU usage is:

- approx. ~~60%~~ 30-40 for MPEG @ 288x160
- approx. 50% for Truemotion @ 640x480

Saturated 320 x 448

Even with no customized hardware assist for decoding!



TrueMotion 2x Decoder

Currently, all TM2X for Dreamcast being shiped decompressors (libraries and codecs) support output via YUV 422 stride formatted textures

The Decompression Library (DXL) provides decompression services for audio and video

TrueMotion 2x based Kamui Examples

Player

✧Plays back a movie

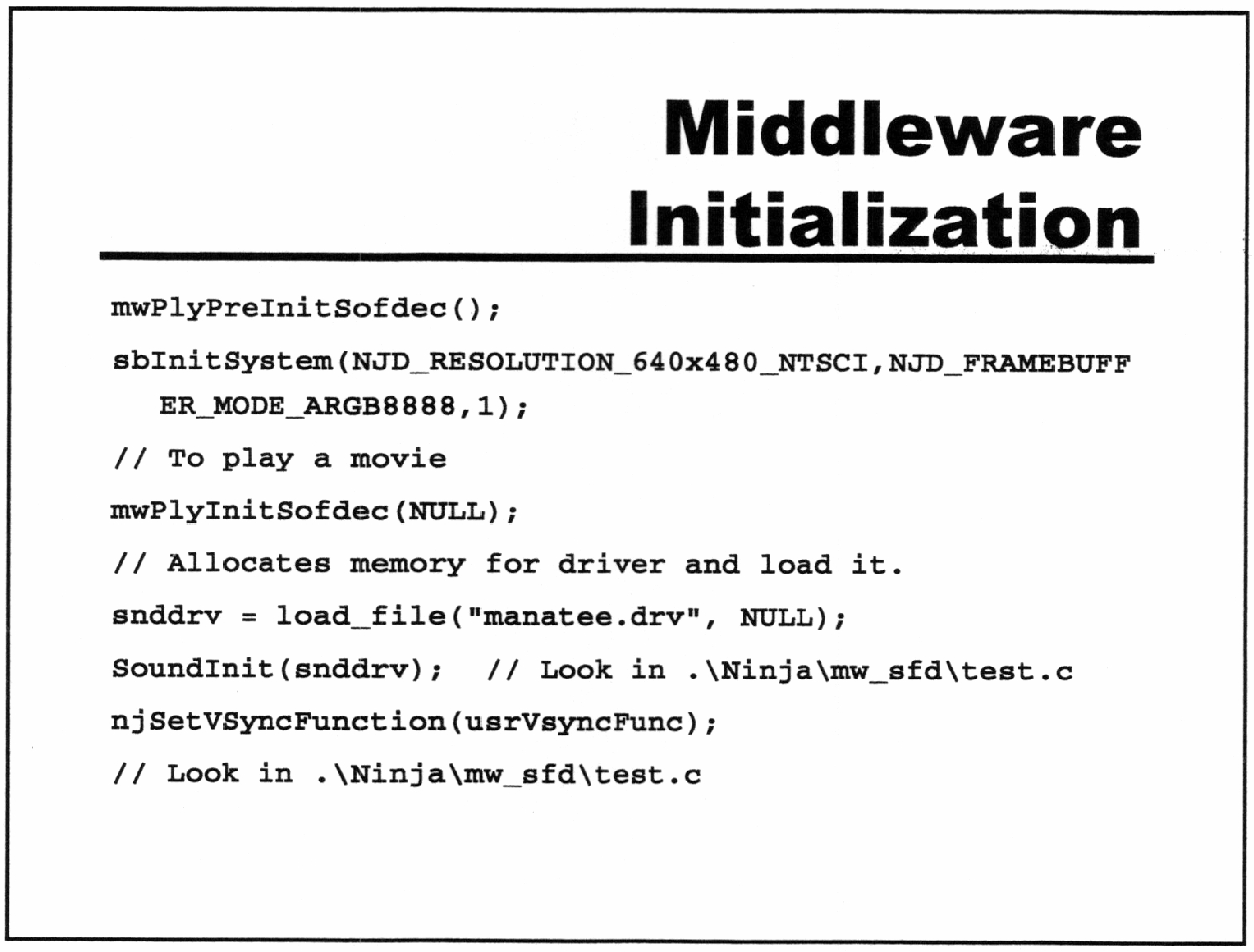
Knot

✧Plays back a movie mapped onto a model

TrueMotion 2x based Kamui Examples

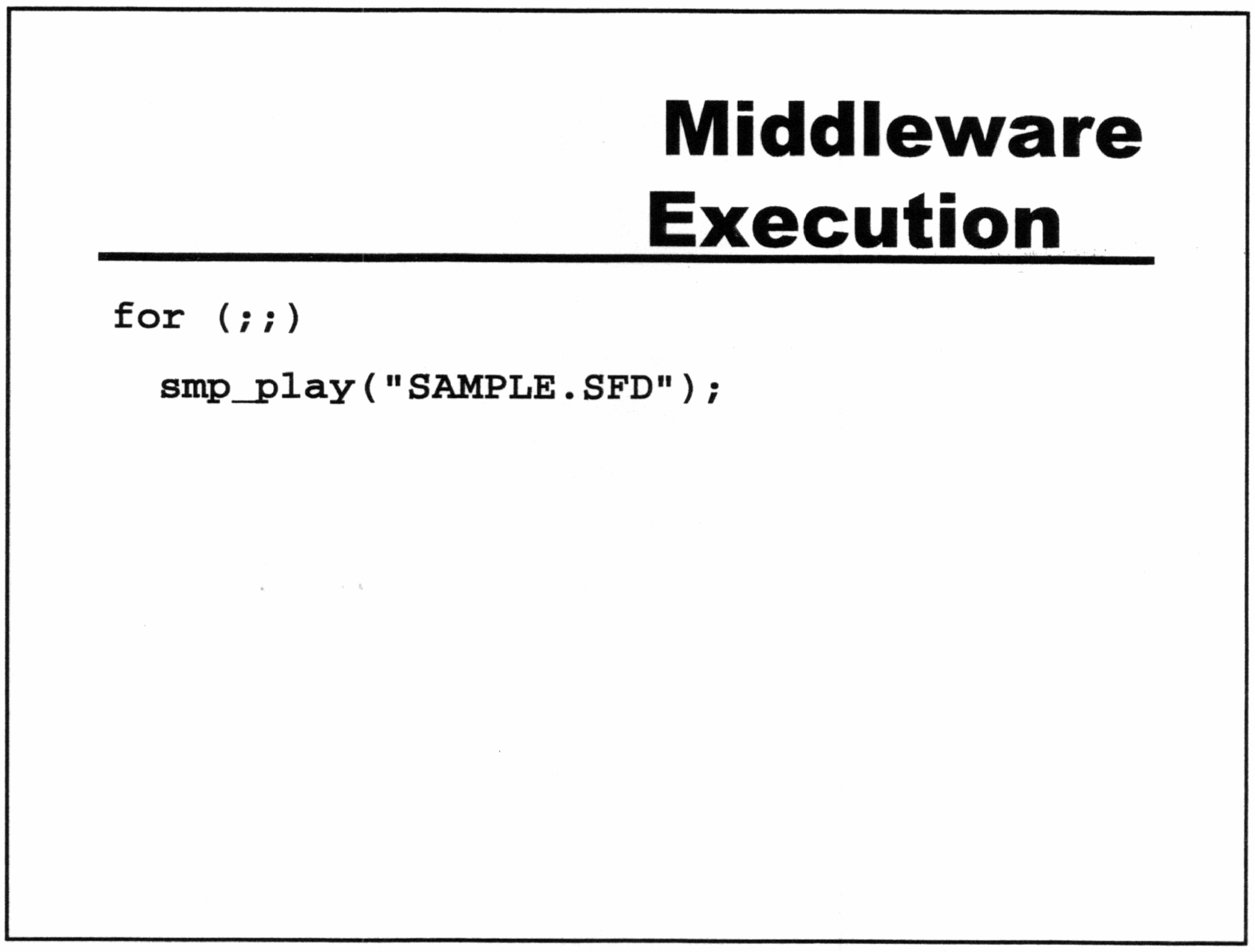
Demonstration

✧Playing back the encoded movie in Duck



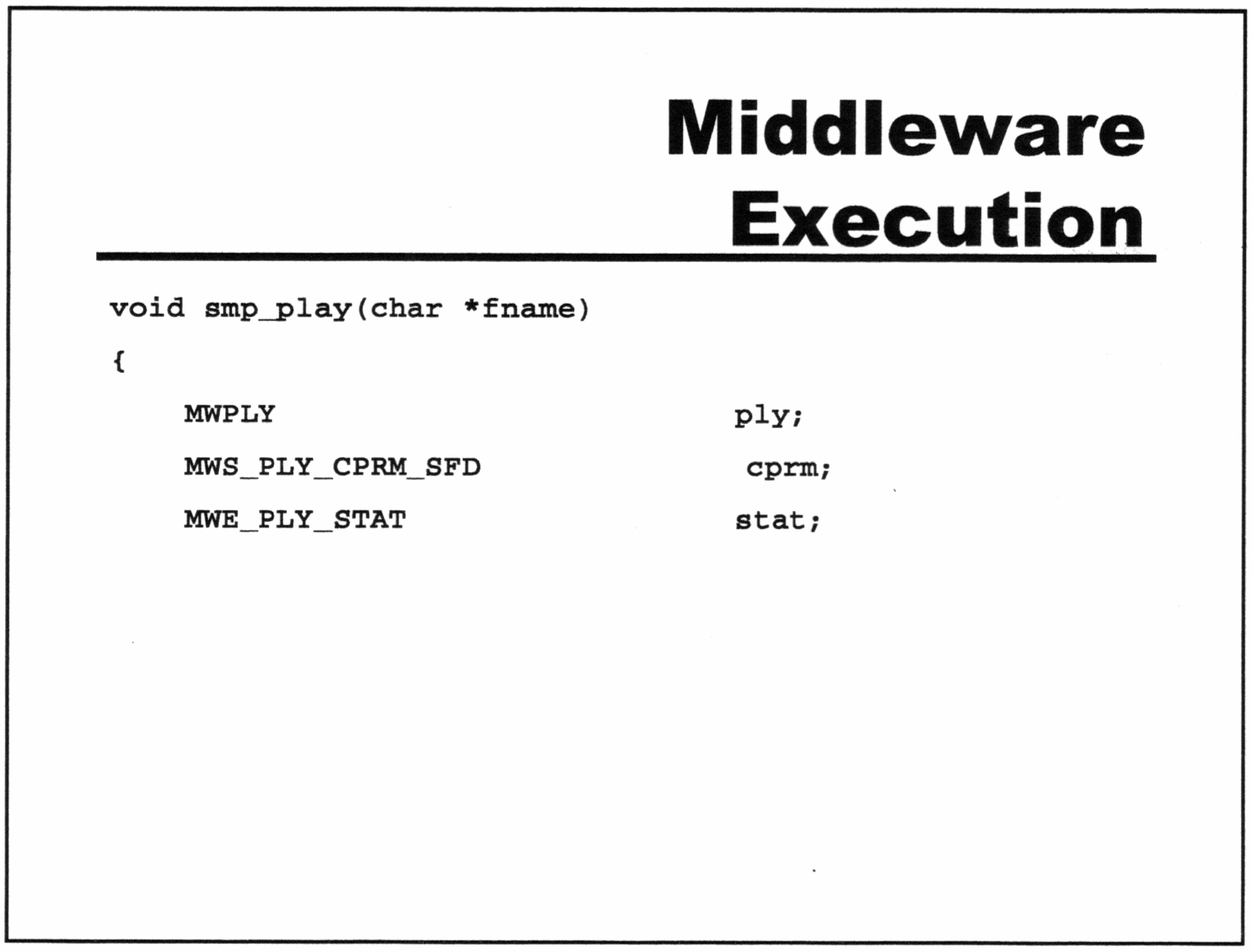
Middleware Initialization

```
mwPlyPreInitSofdec();
sbInitSystem(NJD_RESOLUTION_640x480_NTSCI,NJD_FRAMEBUFF
   ER_MODE_ARGB8888,1);
// To play a movie
mwPlyInitSofdec(NULL);
// Allocates memory for driver and load it.
snddrv = load_file("manatee.drv", NULL);
SoundInit(snddrv);  // Look in .\Ninja\mw_sfd\test.c
njSetVSyncFunction(usrVsyncFunc);
// Look in .\Ninja\mw_sfd\test.c
```



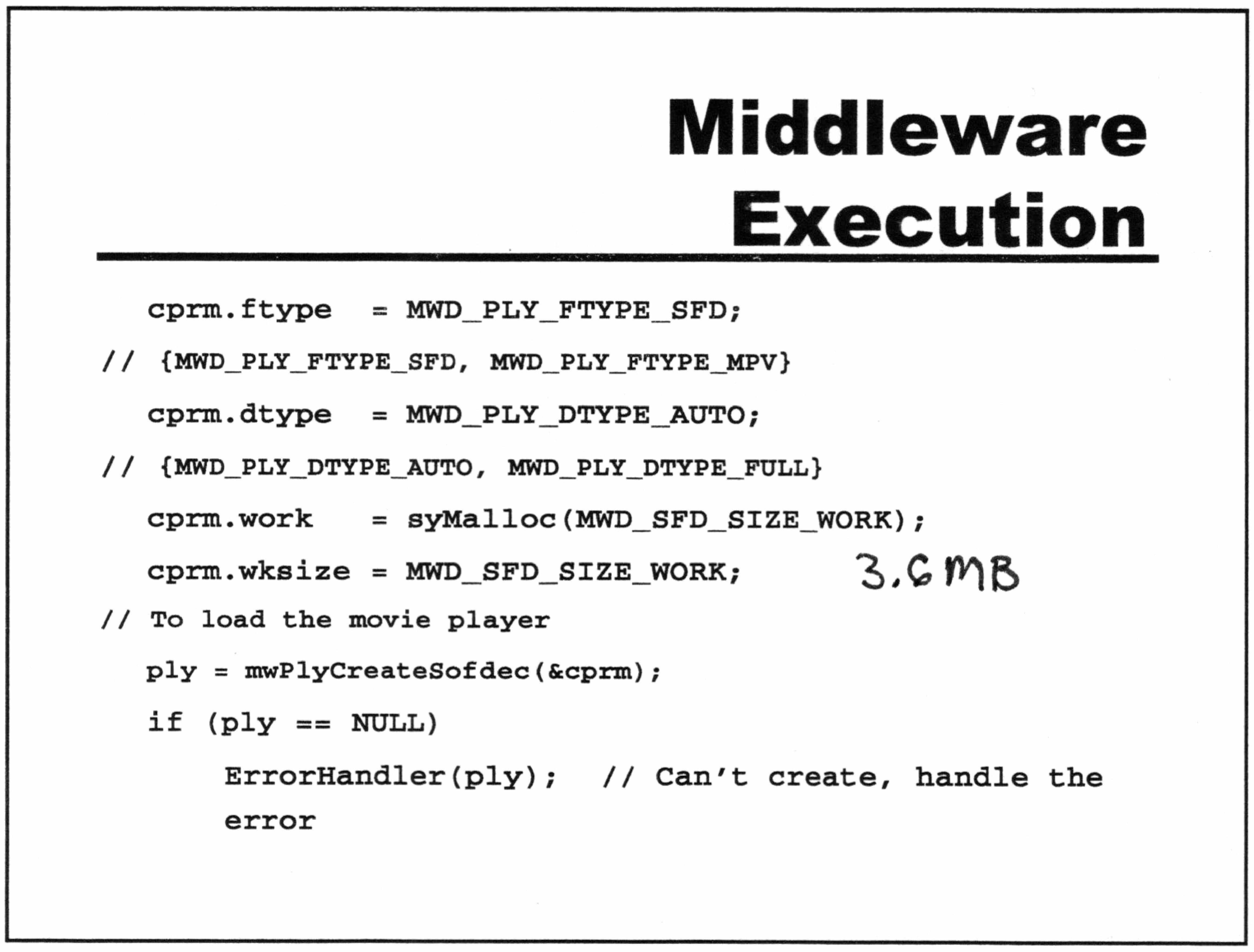
Middleware Execution

```
for (;;)
   smp_play("SAMPLE.SFD");
```



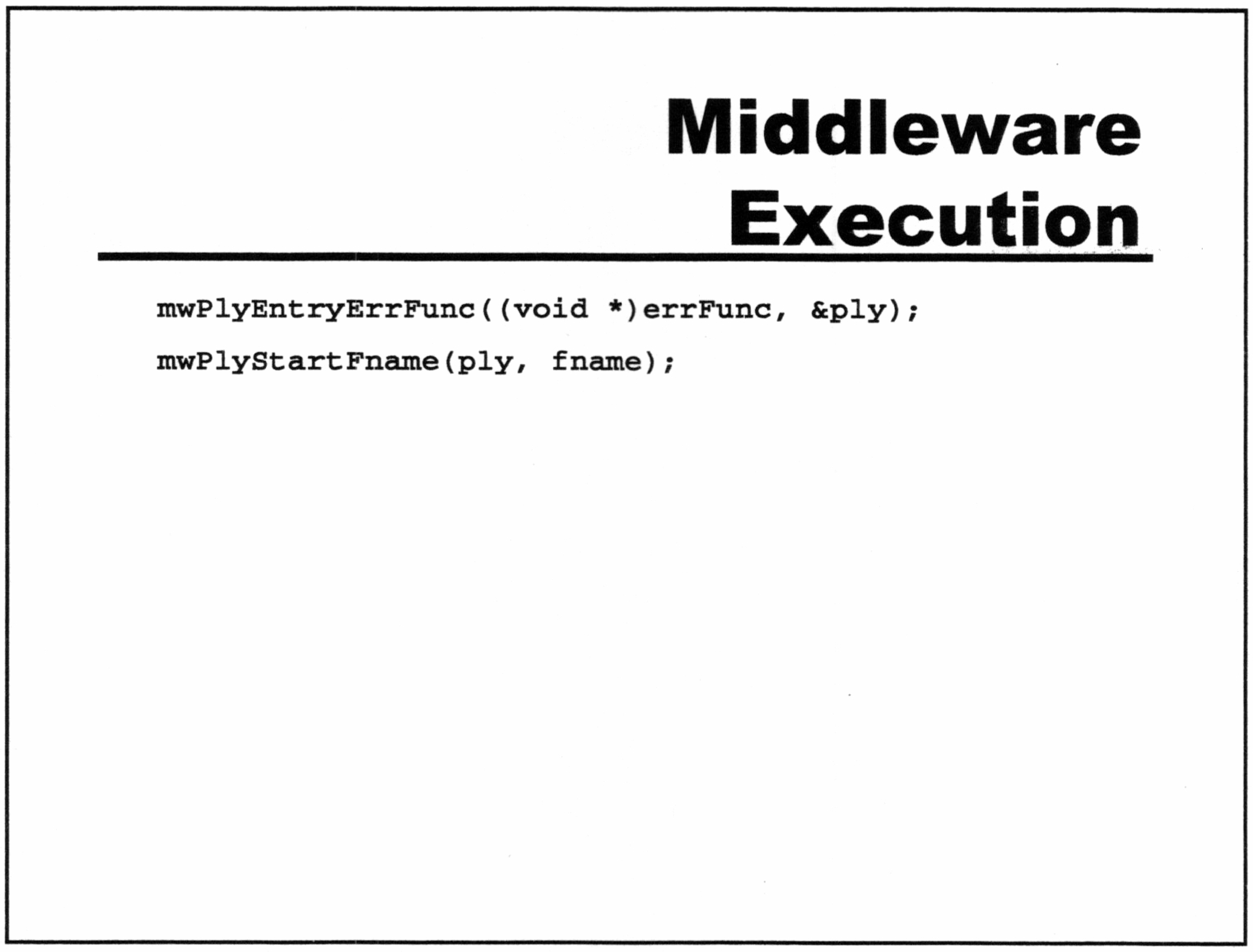
Middleware Execution

```
void smp_play(char *fname)
{
    MWPLY                            ply;
    MWS_PLY_CPRM_SFD                  cprm;
    MWE_PLY_STAT                     stat;
```



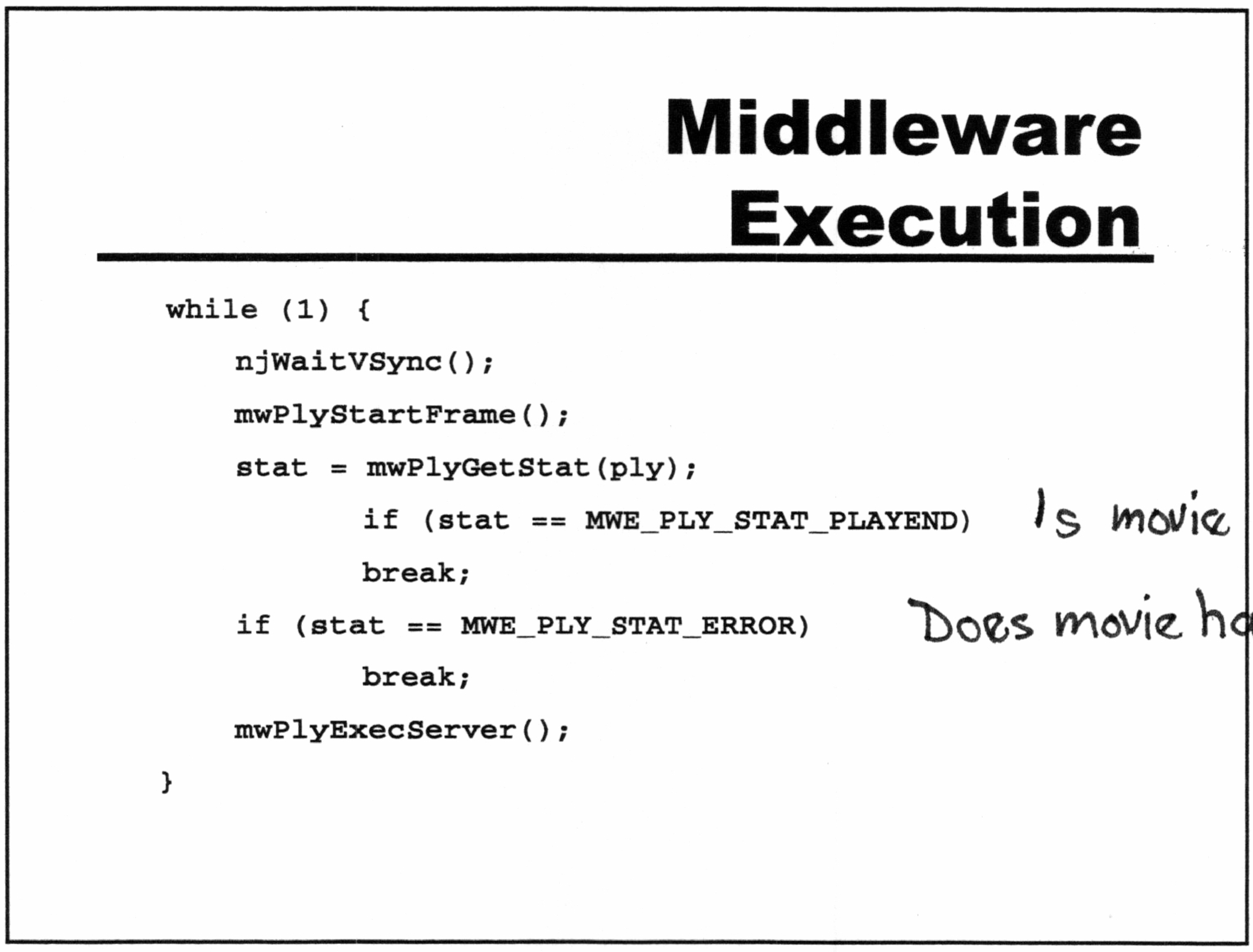
Middleware Execution

```
   cprm.ftype   = MWD_PLY_FTYPE_SFD;
// {MWD_PLY_FTYPE_SFD, MWD_PLY_FTYPE_MPV}
   cprm.dtype   = MWD_PLY_DTYPE_AUTO;
// {MWD_PLY_DTYPE_AUTO, MWD_PLY_DTYPE_FULL}
   cprm.work    = syMalloc(MWD_SFD_SIZE_WORK);
   cprm.wksize = MWD_SFD_SIZE_WORK;        3.6MB
// To load the movie player
   ply = mwPlyCreateSofdec(&cprm);
   if (ply == NULL)
       ErrorHandler(ply);  // Can't create, handle the
       error
```



Middleware Execution

```
mwPlyEntryErrFunc((void *)errFunc, &ply);
mwPlyStartFname(ply, fname);
```

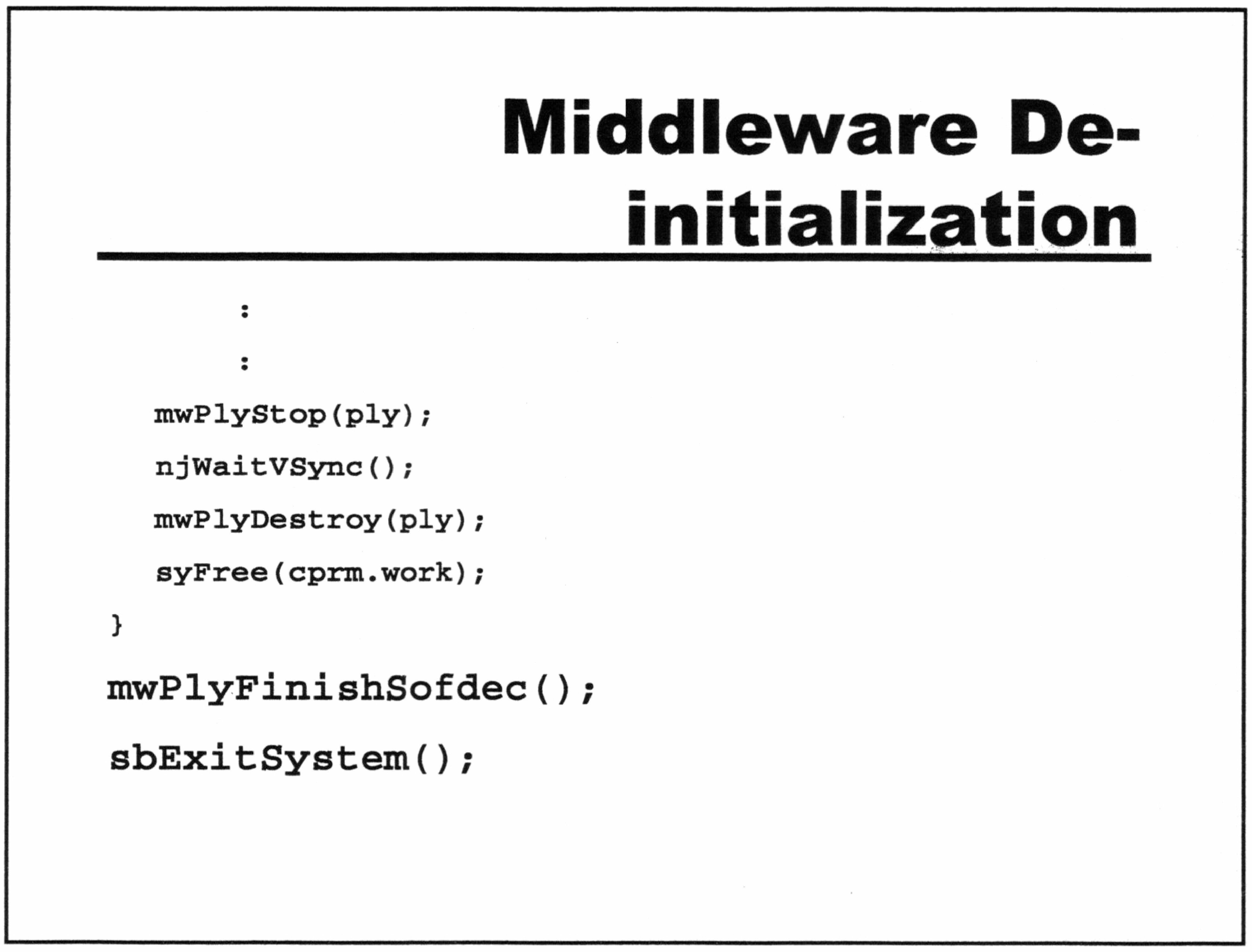


Middleware Execution

```
while (1) {
    njWaitVSync();
    mwPlyStartFrame();
    stat = mwPlyGetStat(ply);
           if (stat == MWE_PLY_STAT_PLAYEND)
           break;
    if (stat == MWE_PLY_STAT_ERROR)
           break;
    mwPlyExecServer();
}
```

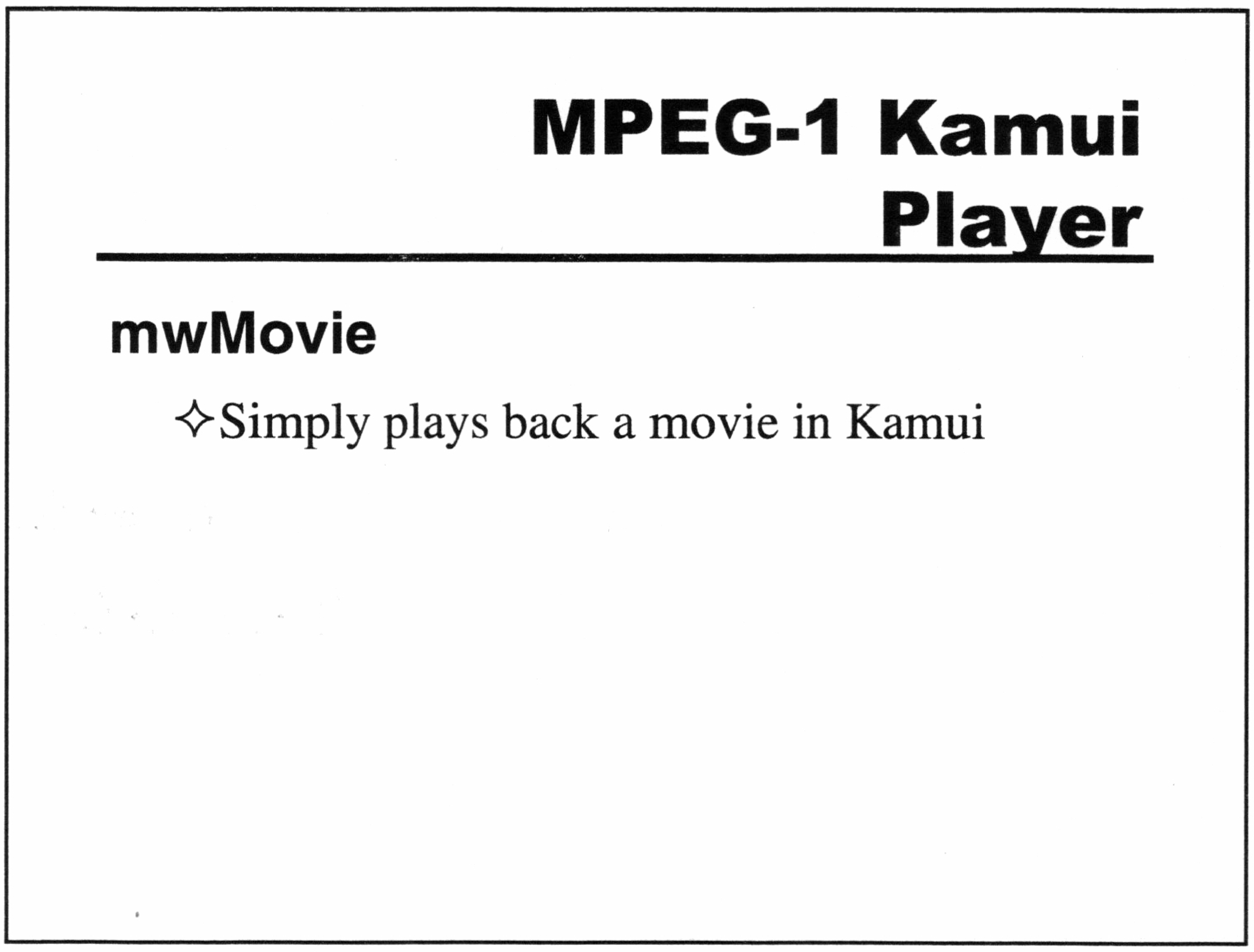
Is movie over?

Does movie have an error?



Middleware De-initialization

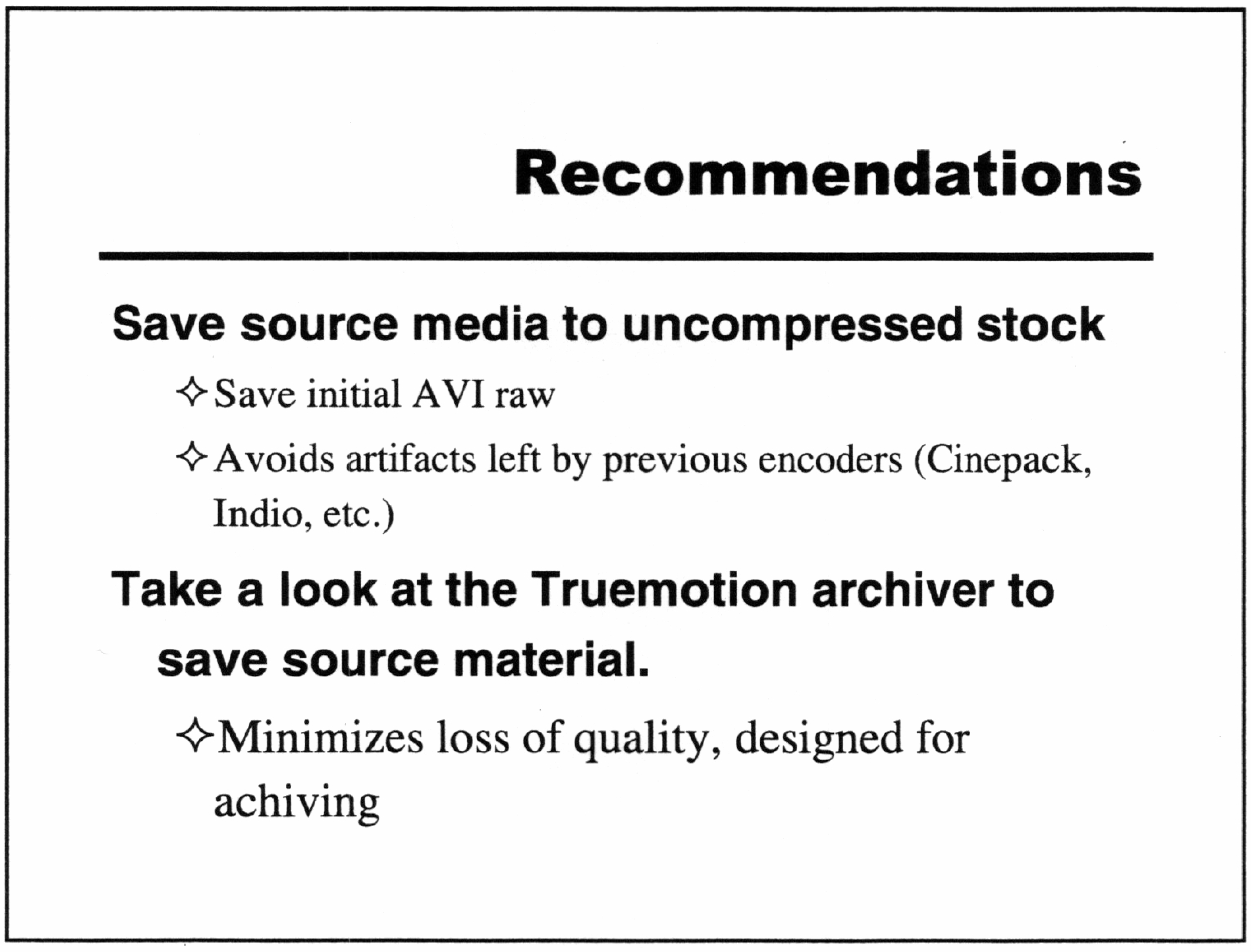
```
        :
        :
    mwPlyStop(ply);
    njWaitVSync();
    mwPlyDestroy(ply);
    syFree(cprm.work);
}
mwPlyFinishSofdec();
sbExitSystem();
```



MPEG-1 Kamui Player

mwMovie

- ✧Simply plays back a movie in Kamui



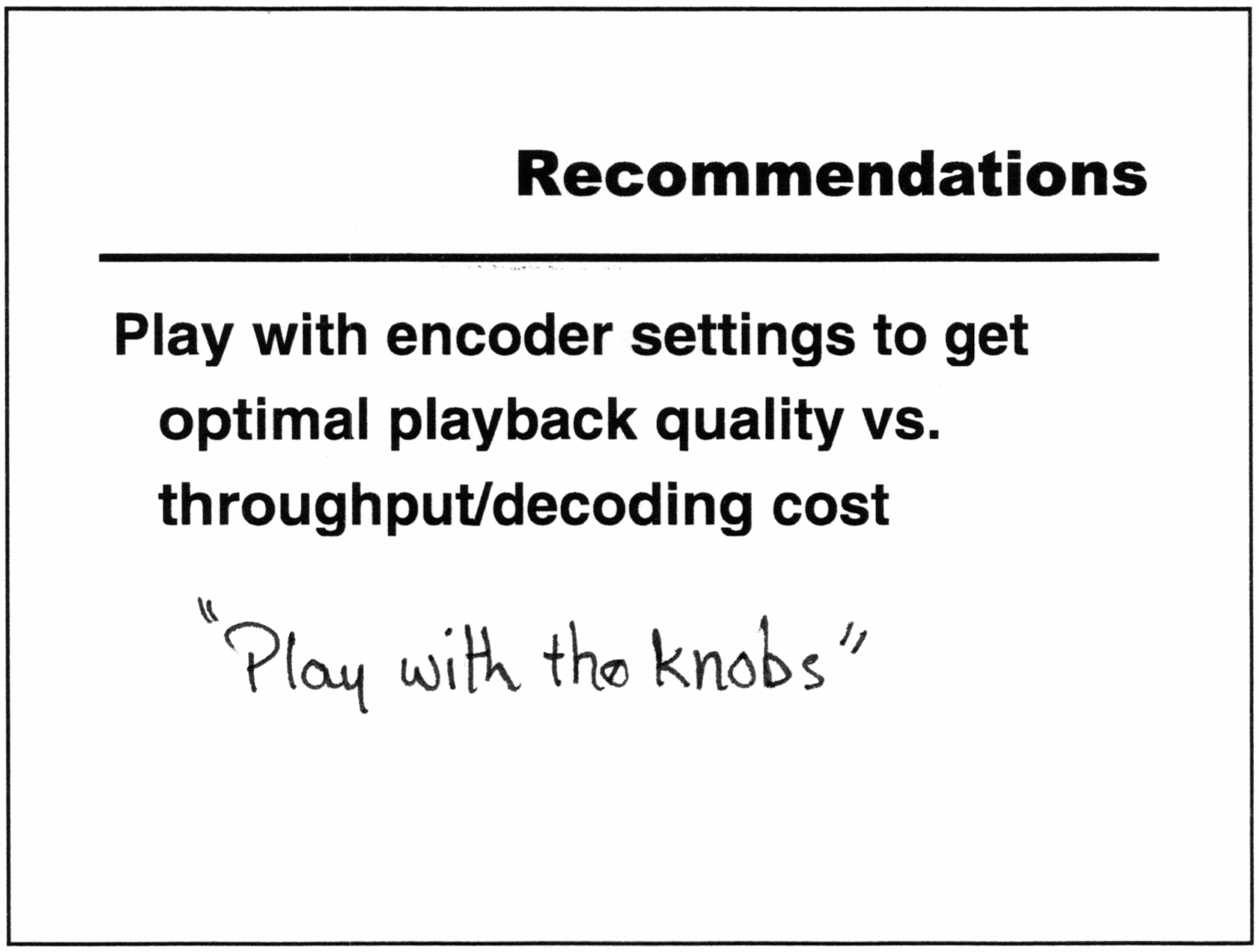
Recommendations

Save source media to uncompressed stock

- ✧Save initial AVI raw
- ✧Avoids artifacts left by previous encoders (Cinepack, Indio, etc.)

Take a look at the Truemotion archiver to save source material.

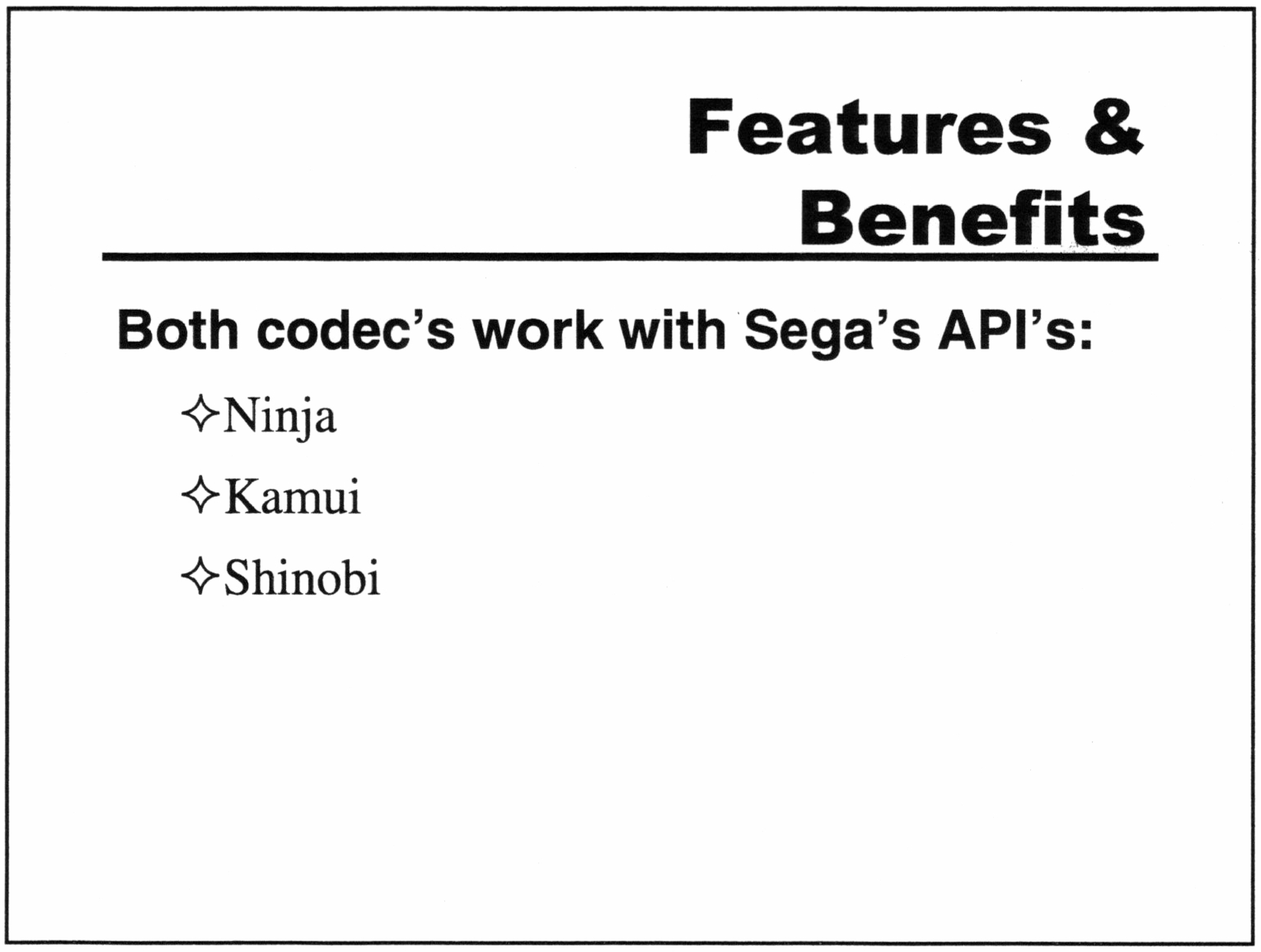
- ✧Minimizes loss of quality, designed for achiving



Recommendations

Play with encoder settings to get optimal playback quality vs. throughput/decoding cost

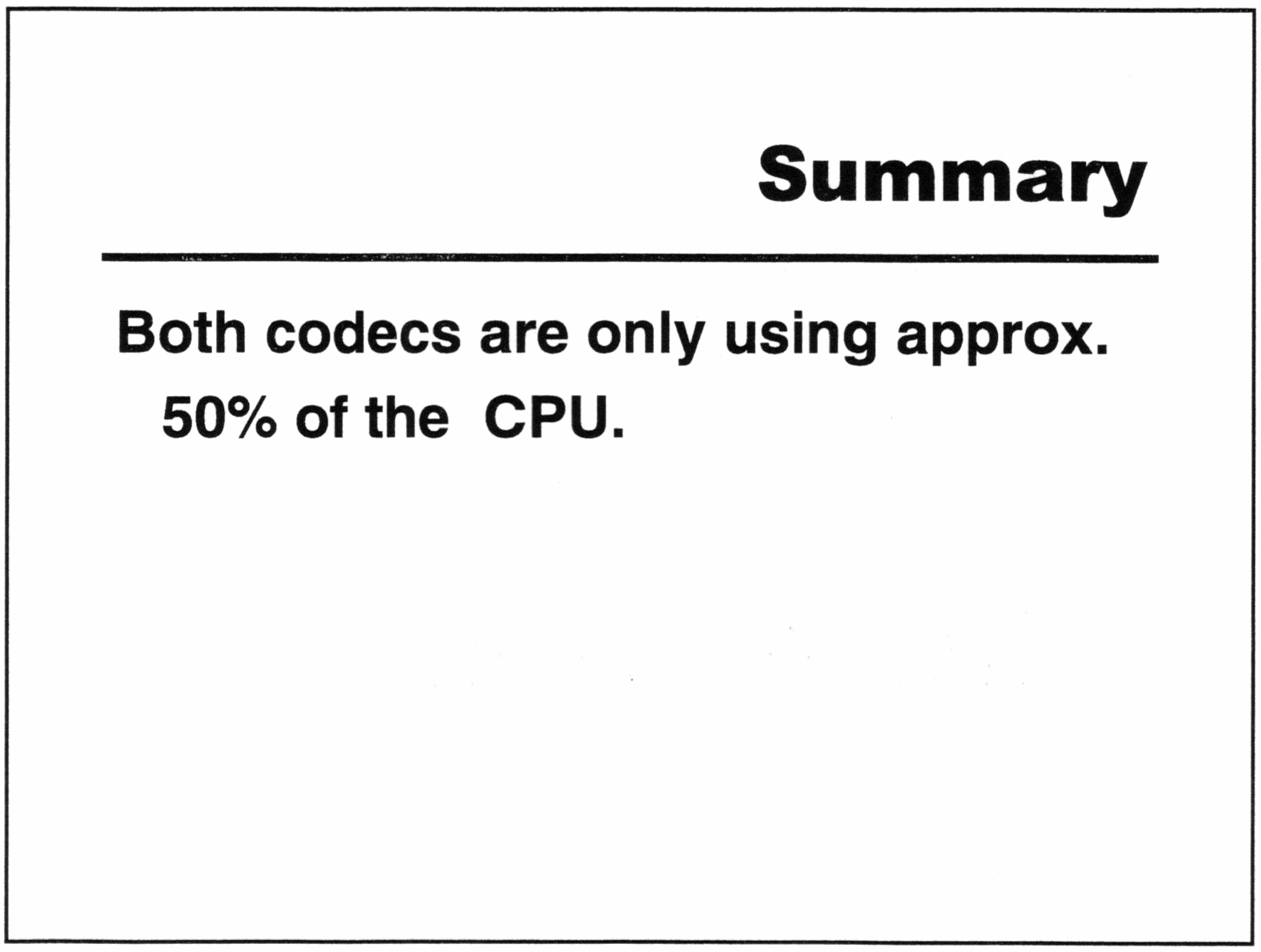
"Play with the knobs"



Features & Benefits

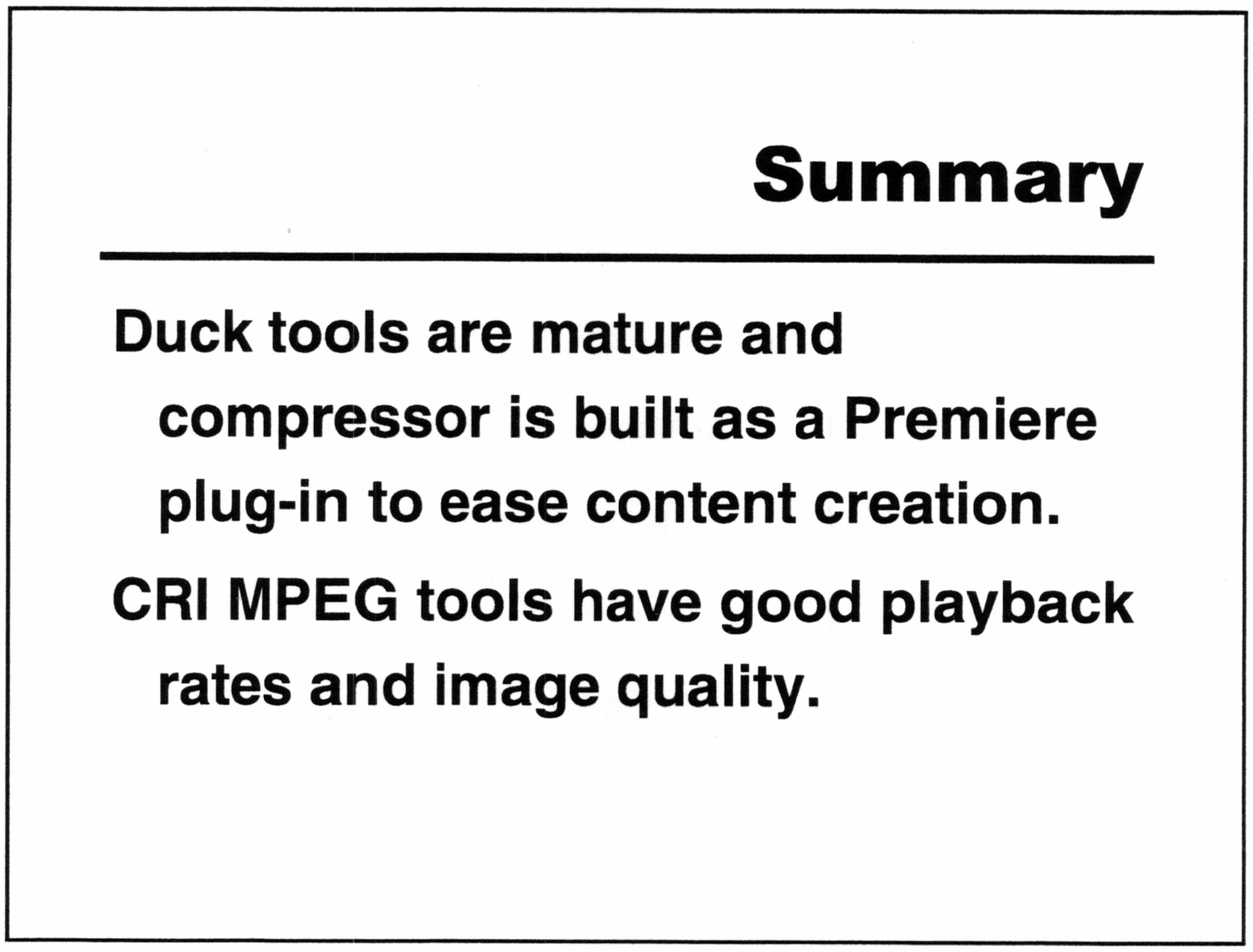
Both codec’s work with Sega’s API’s:

- ✧Ninja
- ✧Kamui
- ✧Shinobi



Summary

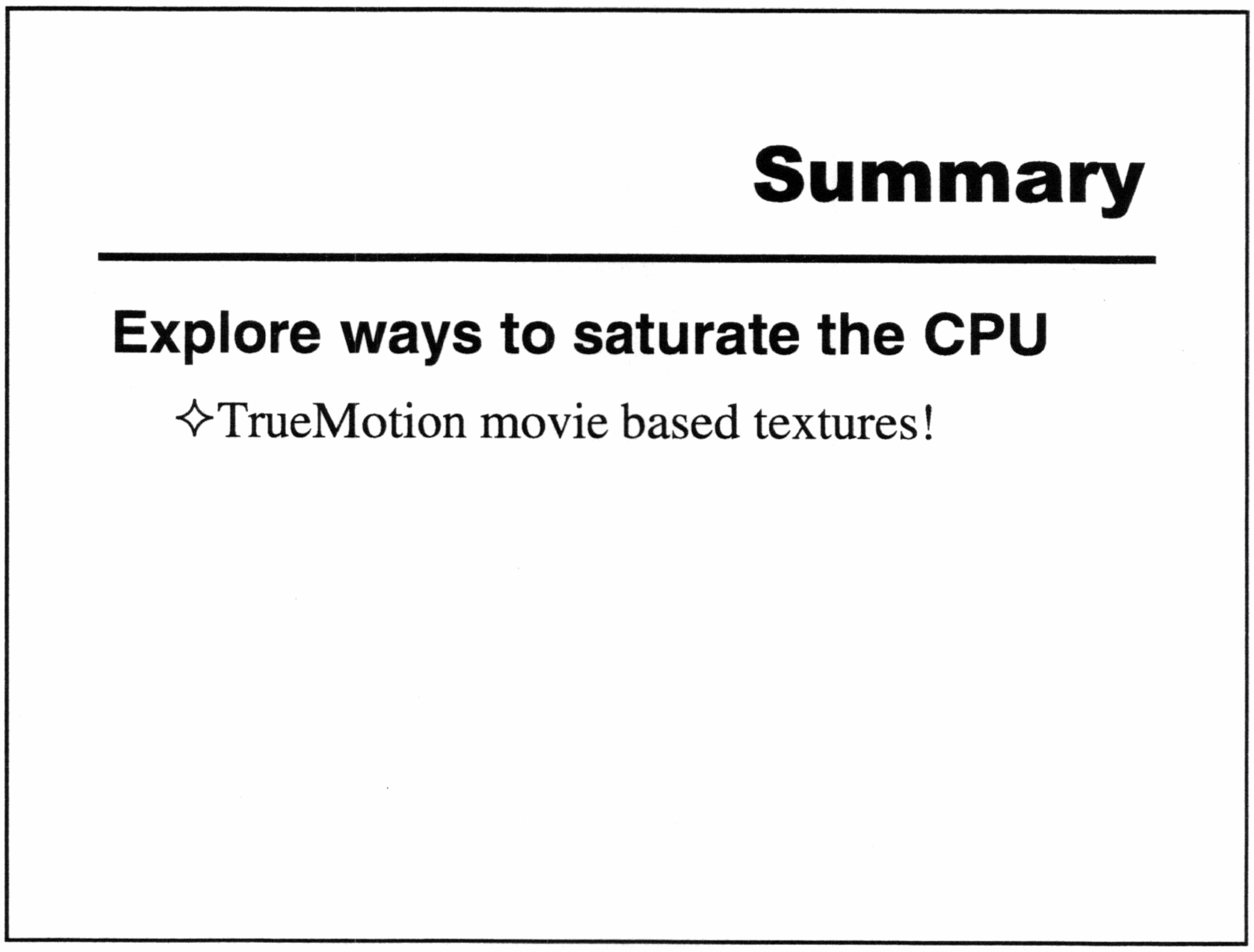
Both codecs are only using approx. 50% of the CPU.



Summary

Duck tools are mature and compressor is built as a Premiere plug-in to ease content creation.

CRI MPEG tools have good playback rates and image quality.



Summary

Explore ways to saturate the CPU

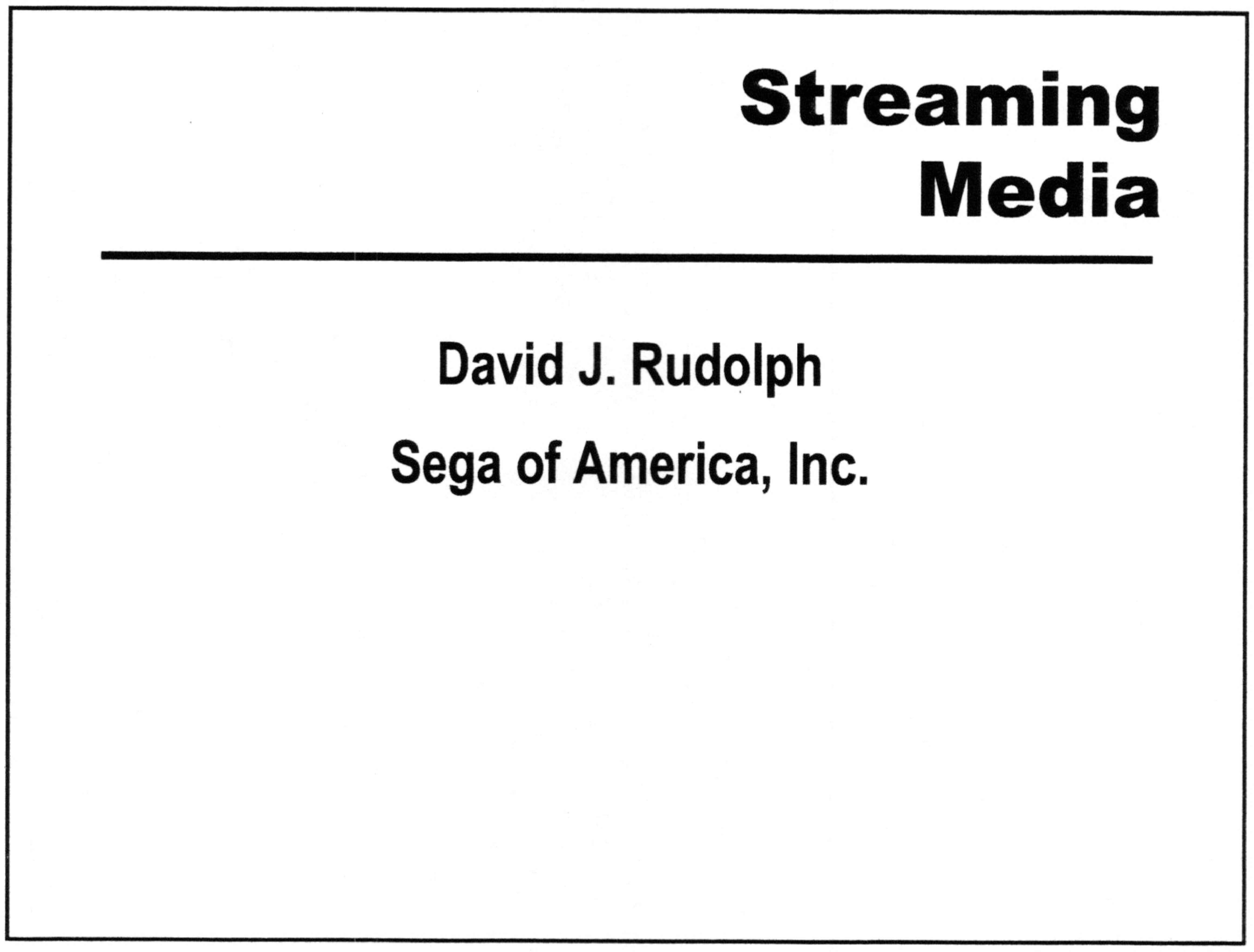
- ✧TrueMotion movie based textures!

Availability

Will be on Sega U.S. 7.0 SDK due at the end of April.

Available upon request

Questions



Streaming Media

David J. Rudolph

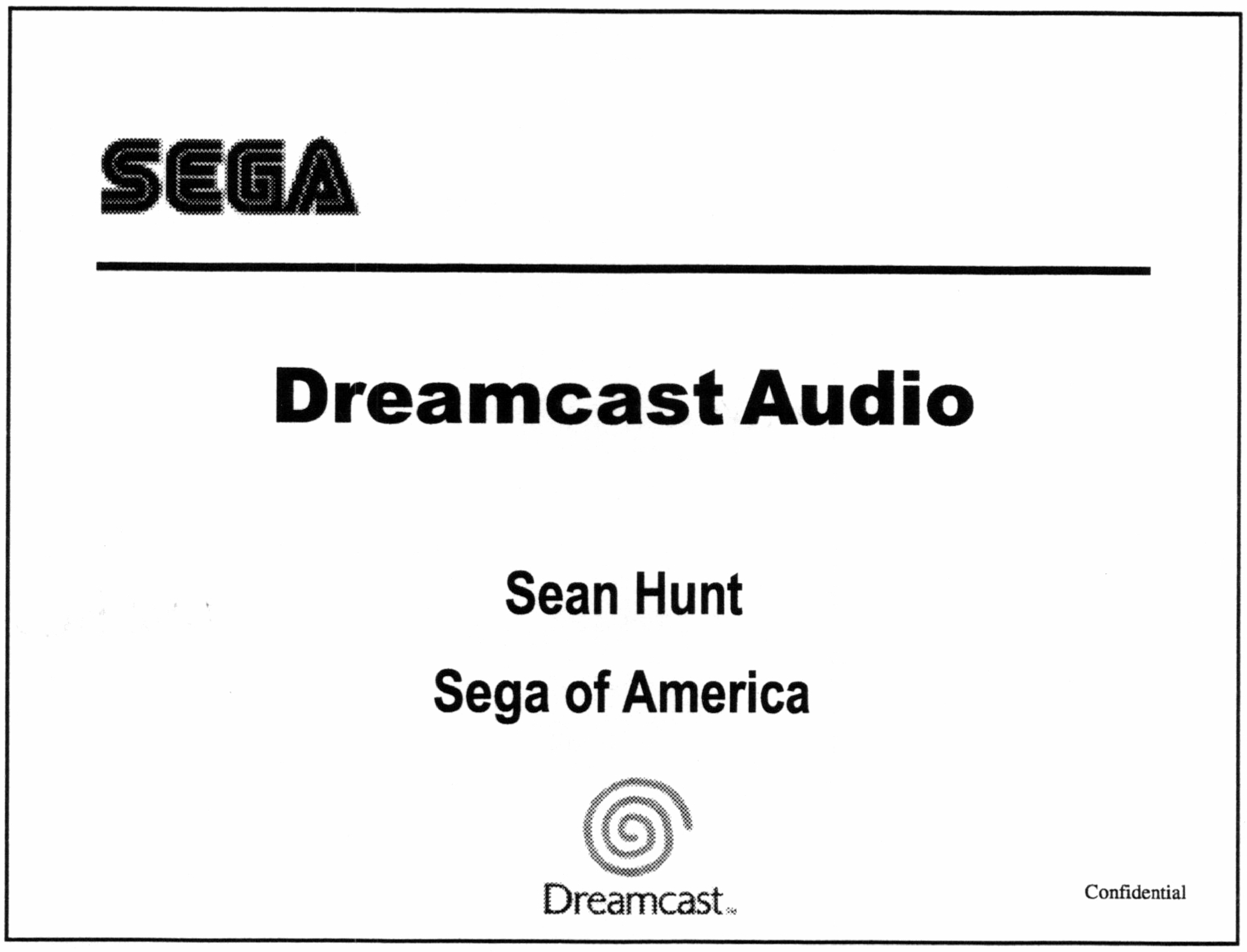
Sega of America, Inc.

SEGA

Dreamcast Audio

Sean Hunt
Sega of America

Dreamcast™



SEGA

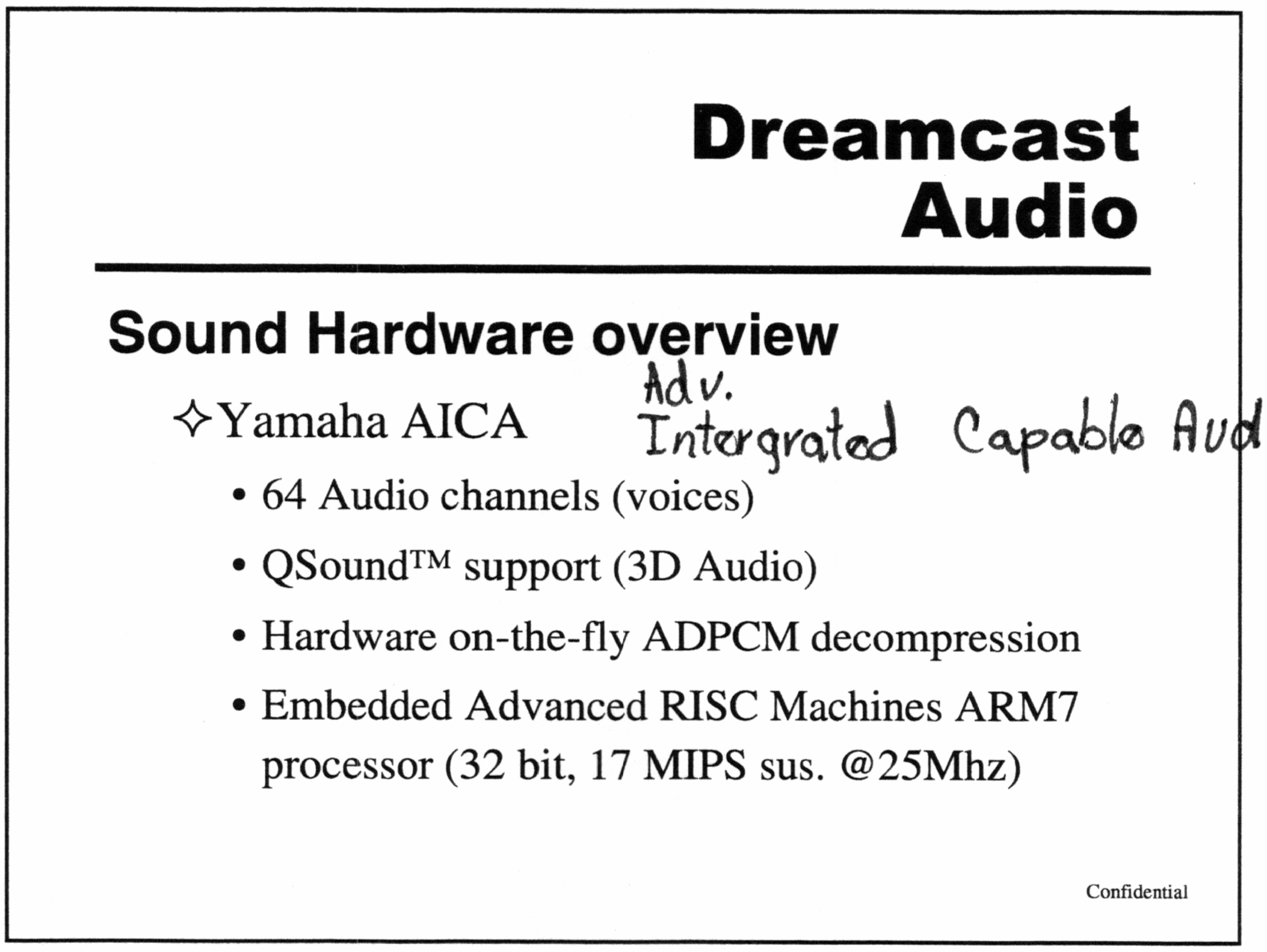
Dreamcast Audio

Sean Hunt

Sega of America

Dreamcast

Confidential



Dreamcast Audio

Sound Hardware overview

✧Yamaha AICA Adv. Intergrated Capable Audio

- 64 Audio channels (voices)
- QSound™ support (3D Audio)
- Hardware on-the-fly ADPCM decompression
- Embedded Advanced RISC Machines ARM7 processor (32 bit, 17 MIPS sus. @25Mhz)

Confidential

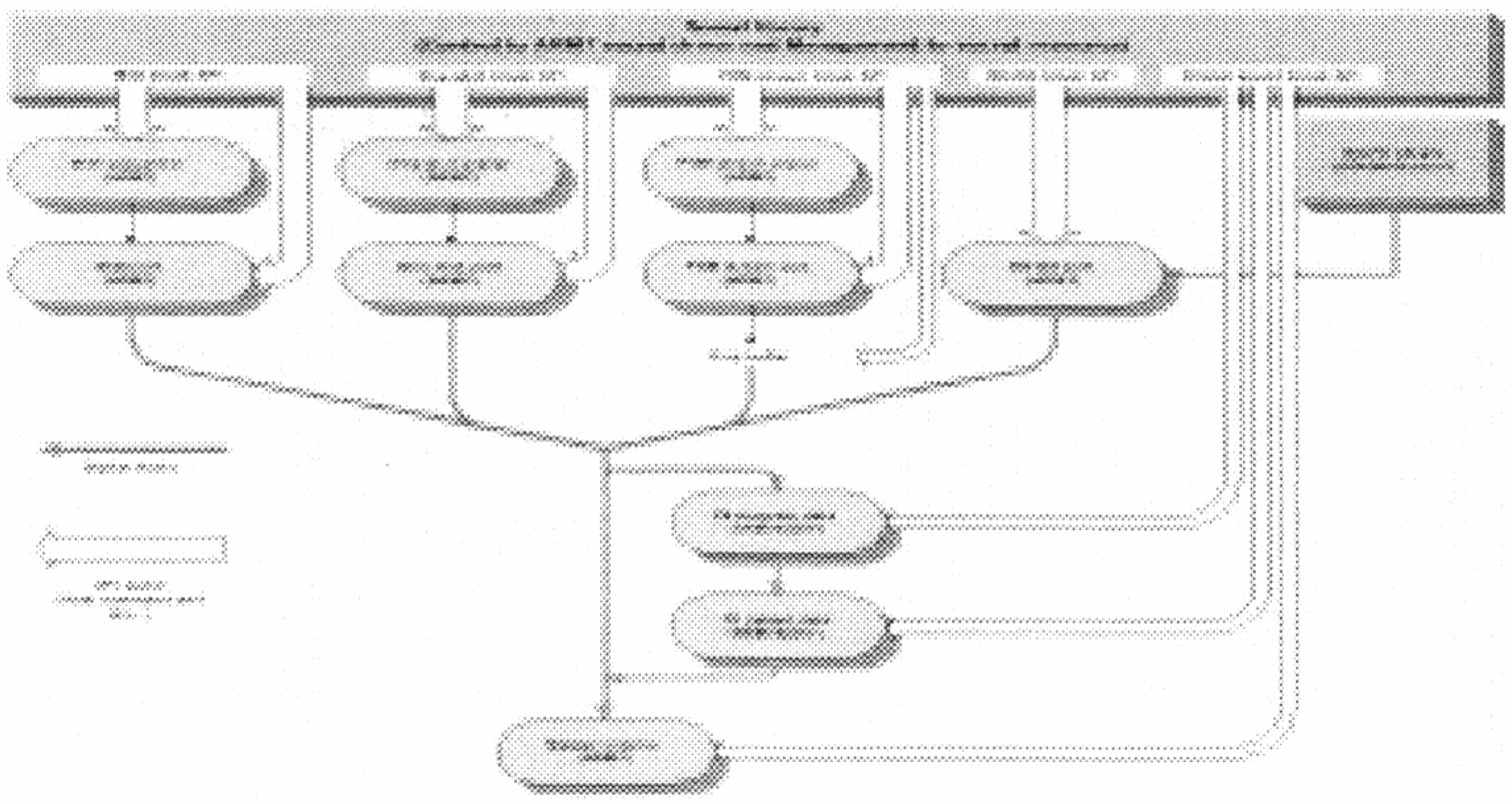
Sound Hardware

- Embedded DSP
  - – 128 step DSP
  - – 24 bit internal sampling rate
  - – 10 MIPS
  - – 16 Digital Inputs. 16 Digital outs
  - – 64 LPFs with envelopes (one per channel) Low Pass Filters
  - – 2 LFOs per channel (pitch and amplitude)
  - – Algorithms can be connected serially or in parallel

Confidential

Dreamcast Audio

Data flow



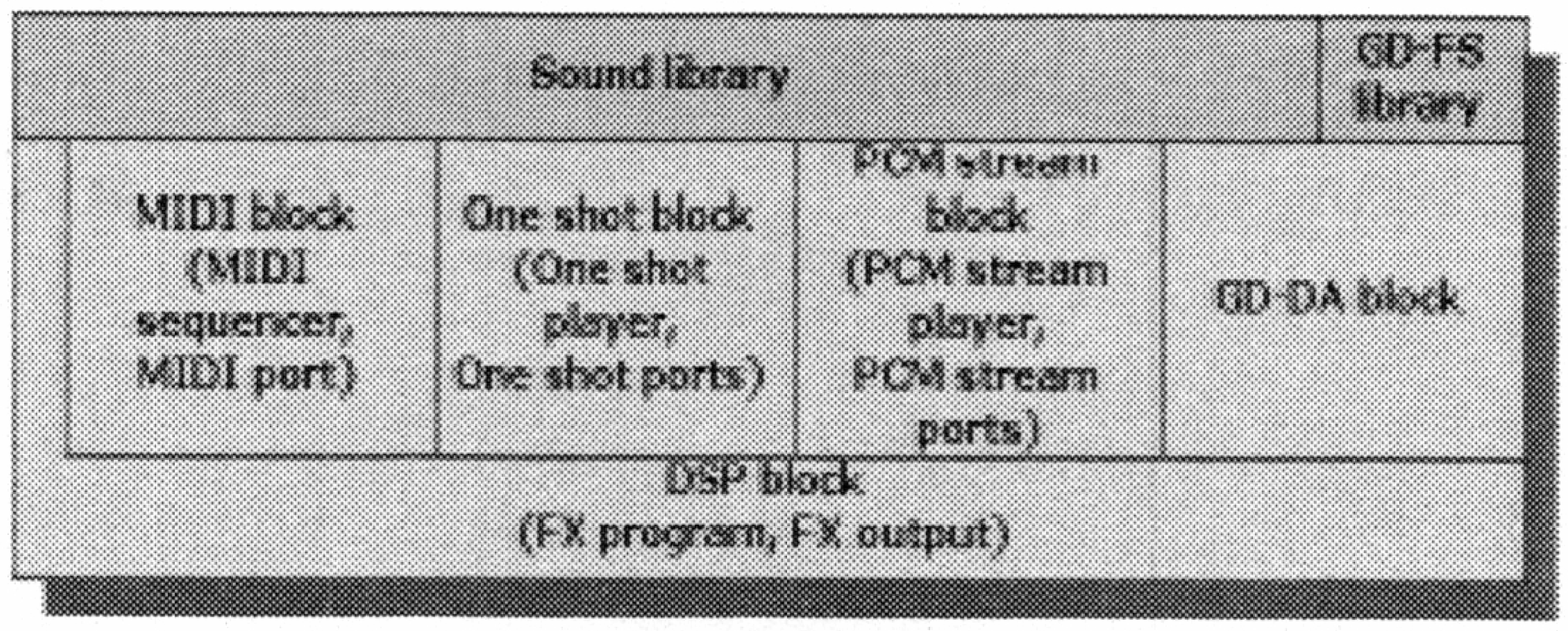
Confidential

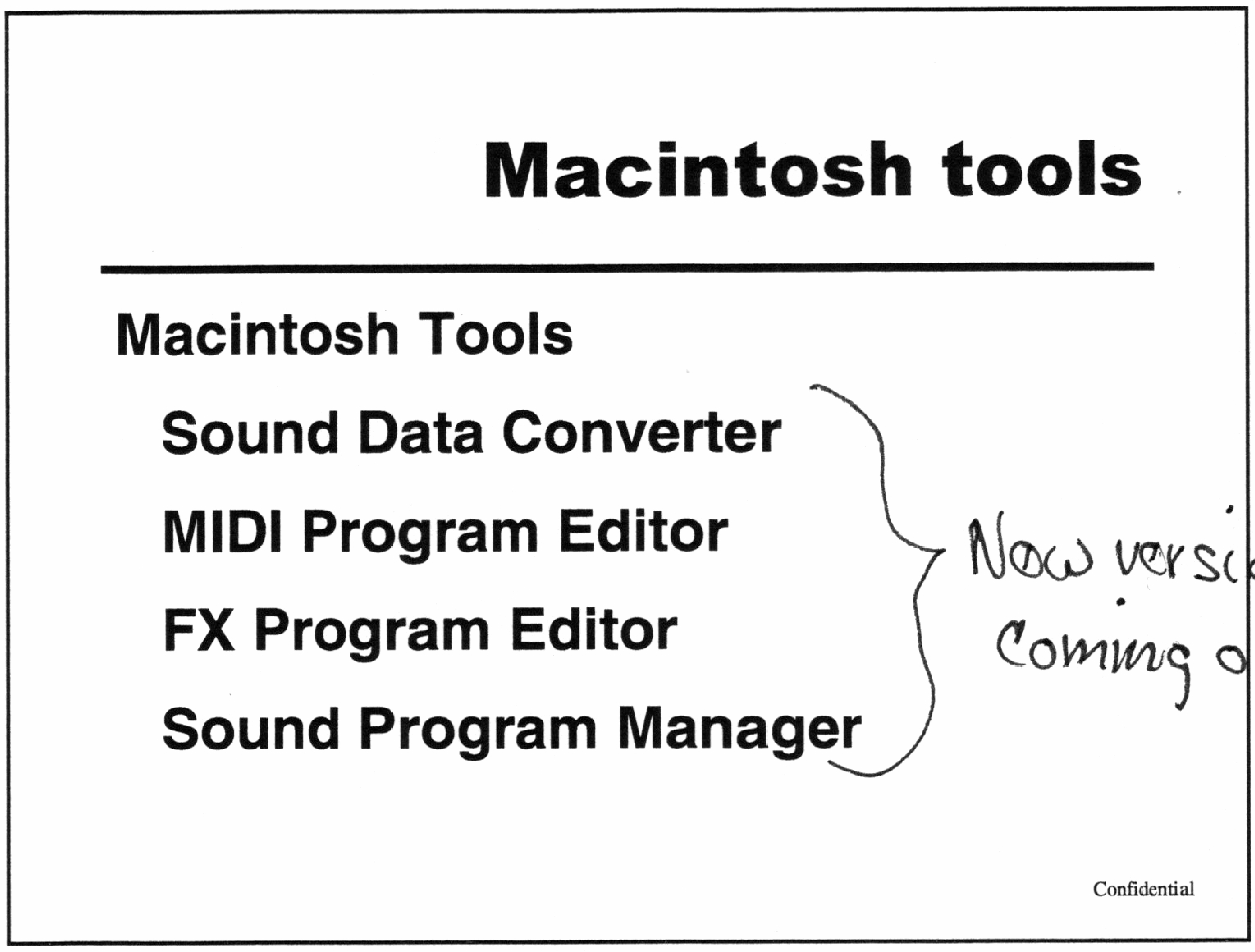
Dreamcast Audio

	MIDI	One-Shot	PCM Stream	GD-DA
Max. Voices	48 (common to all ports)	8	8	1
Max. Port No.	8	8	4 (can mix stereo/mono)	1 (stereo)
Volume setting	Can set for each port		Can set for each channel	
Panpot setting	Can set for each port		Can set for each channel	
Pitch setting	Can set for each port	Can set for each port (affected by speed setting)		Can set for each port
Speed setting	Can set for each port	Can set for each port (affected by pitch setting)		Can set for each port
FX Channel setting	MIDI program data dependent	Can set for each port		FX program data dependent
FX Level setting	Can set for each port			FX program data dependent
Direct Level setting	Can set for each port			FX program data dependent
File Extensions (refer to Terminology)	.msb, .mpb	.osb	.p04, .p08, .p16	No requirement

New driver can init more than 8 ch per section

Dreamcast Audio


Sound library
GD-FS library
MIDI block (MIDI sequencer, MIDI port)
One shot block (One shot player, One shot ports)
PCM stream block (PCM stream player, PCM stream ports)
GD-DA block
DSP block (FX program, FX output)



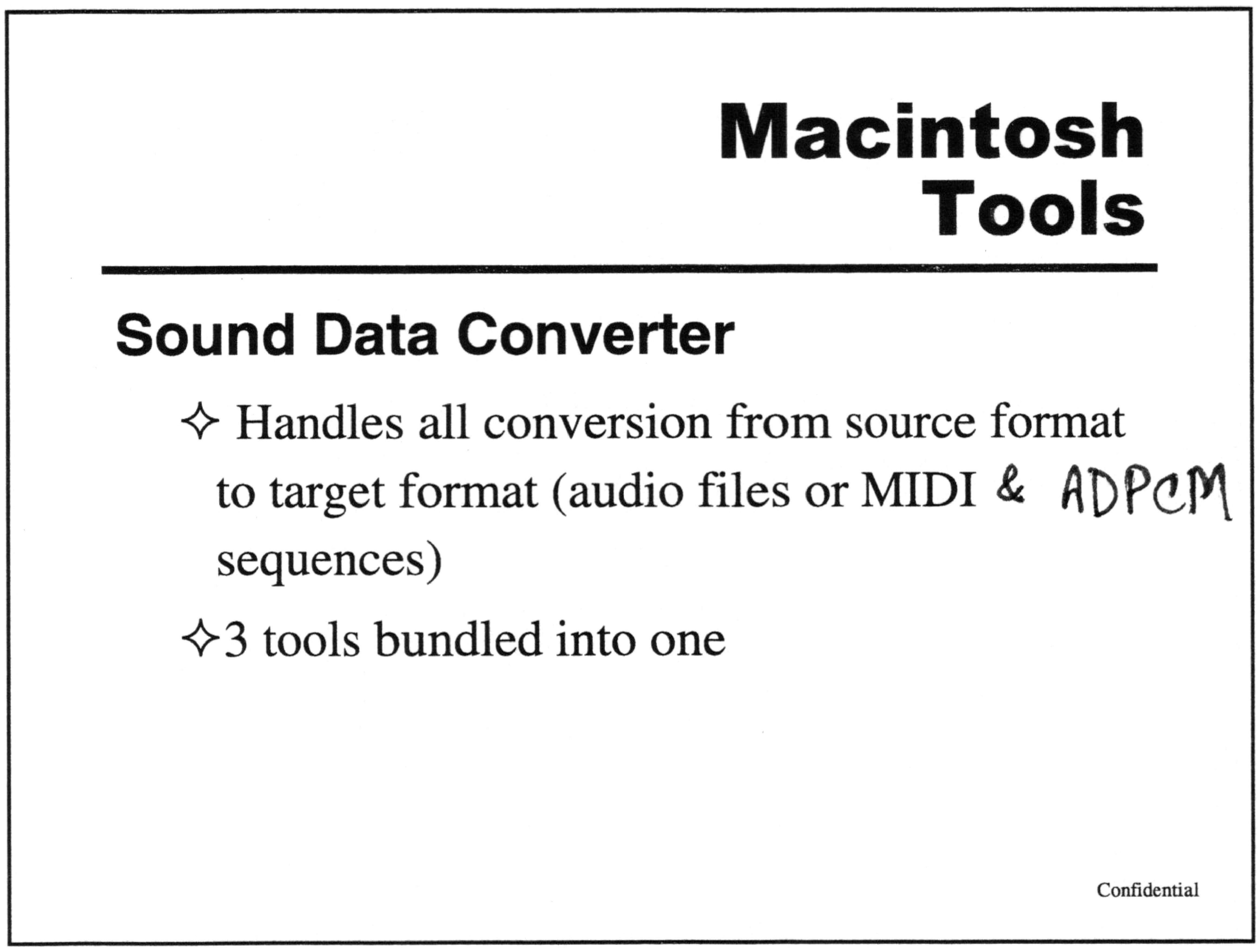
Macintosh tools

Macintosh Tools

- Sound Data Converter
- MIDI Program Editor
- FX Program Editor
- Sound Program Manager

Now versions coming out

Confidential



Macintosh Tools

Sound Data Converter

- ✧ Handles all conversion from source format to target format (audio files or MIDI & ADPCM sequences)
- ✧3 tools bundled into one

Confidential

Macintosh Tools

Sound Data Converter

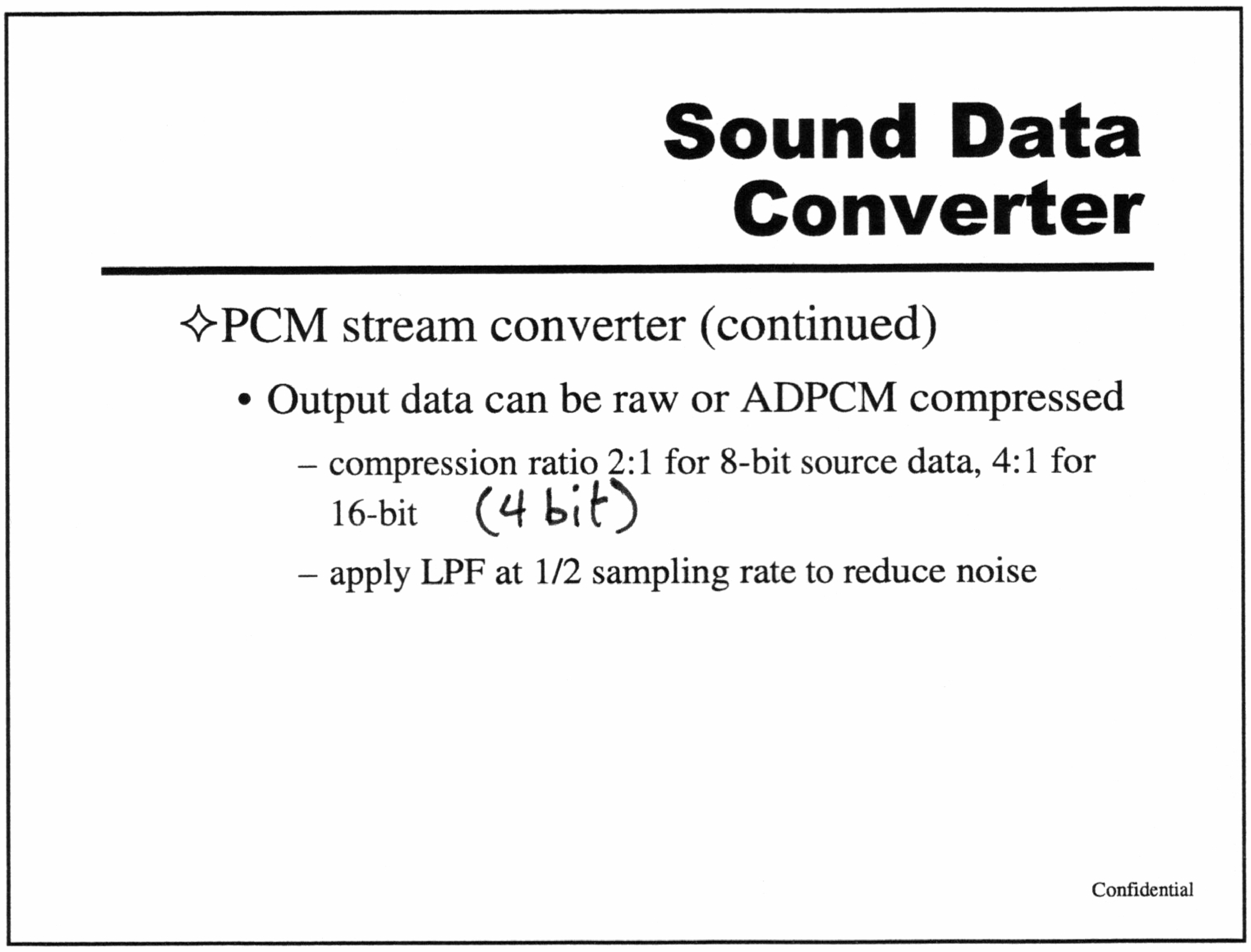
- ✧ PCM stream converter
- ✧One-Shot Bank converter
- ✧MIDI Sequence Bank converter

Confidential

Sound Data Converter

- ✧PCM stream converter
  - Accepts AIFF, SD2, or WAV format audio files
  - Separate into separate left and right streams
  - Source data can be 4-bit, 8-bit, or 16-bit; 11.025, 22.5, or 44.1KHz
    - any extra information (such as loop points) will be stripped out

Confidential

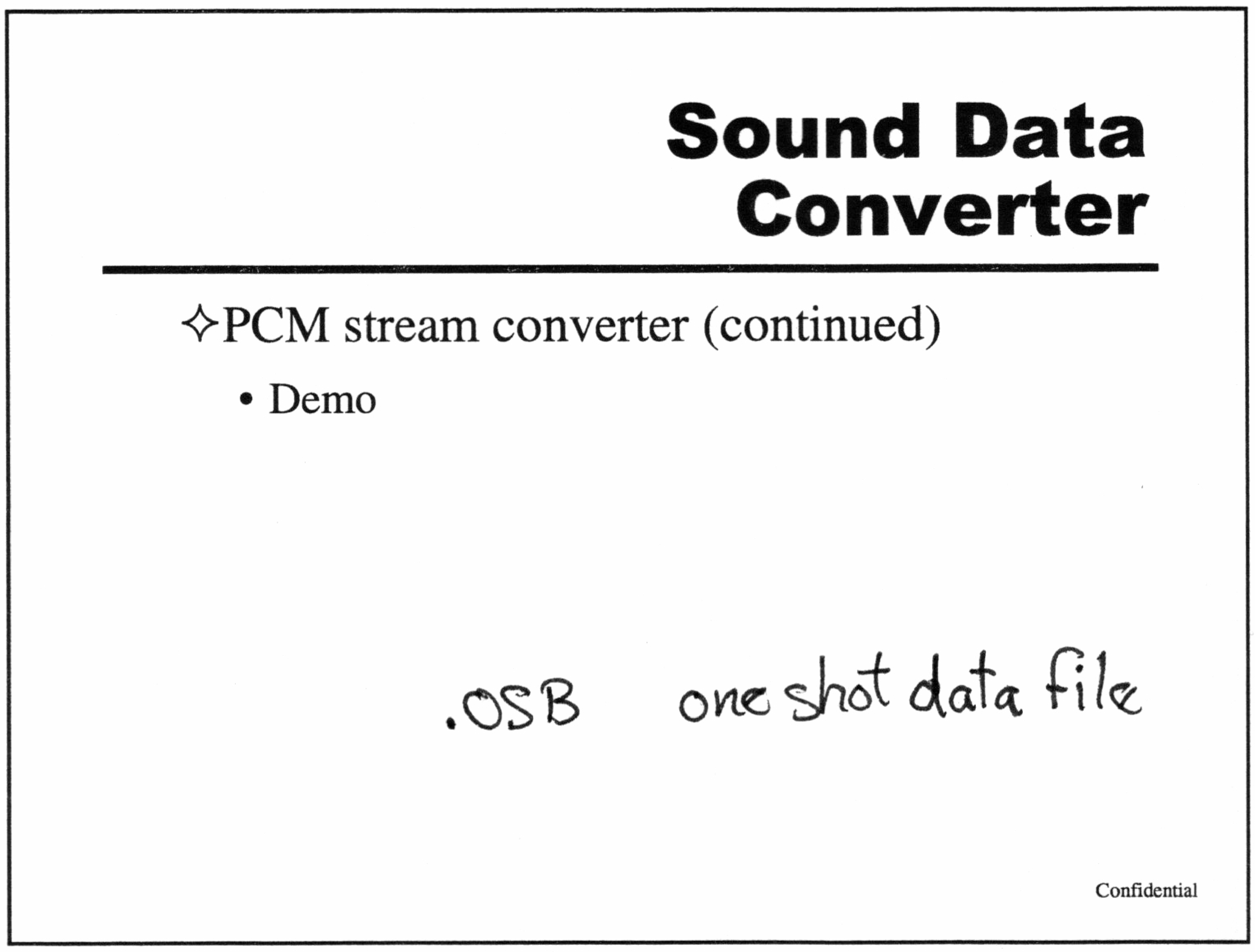


Sound Data Converter

✧PCM stream converter (continued)

- Output data can be raw or ADPCM compressed
  - – compression ratio 2:1 for 8-bit source data, 4:1 for 16-bit (4 bit)
  - – apply LPF at 1/2 sampling rate to reduce noise

Confidential



Sound Data Converter

✧PCM stream converter (continued)

- Demo

.OSB one shot data file

Confidential

Sound Data Converter

✧One-Shot data converter

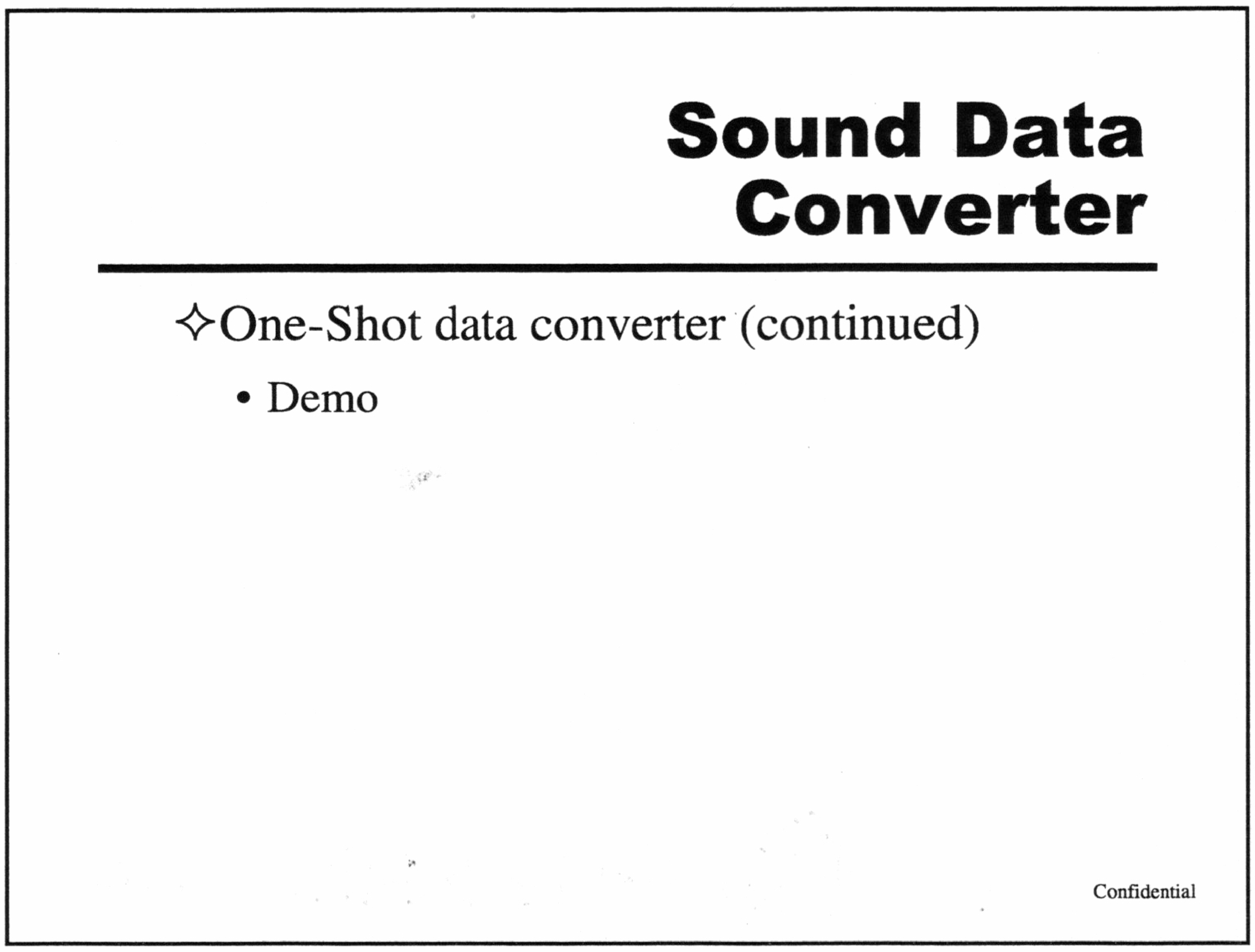
- Accepts AIFF, SD2, or WAV format audio files
- Mono only, < 65534 samples
  - reduce sampling rate to get longer sounds
- Source data can be 4-bit, 8-bit, or 16-bit; 11.025, 22.5, or 44.1KHz
  - loop points are preserved if present, and do not need to be specially aligned

One shot file can not be > 64K samples

Sound Data Converter

✧One-Shot data converter (continued)

- Output data is one-shot bank file (.osb) that is used in Sound Project Manager
  - compression ratio 2:1 for 8-bit source data, 4:1 for 16-bit (this is for memory savings only - does not allow larger samples)
  - apply LPF at 1/2 sampling rate to reduce noise due to compression

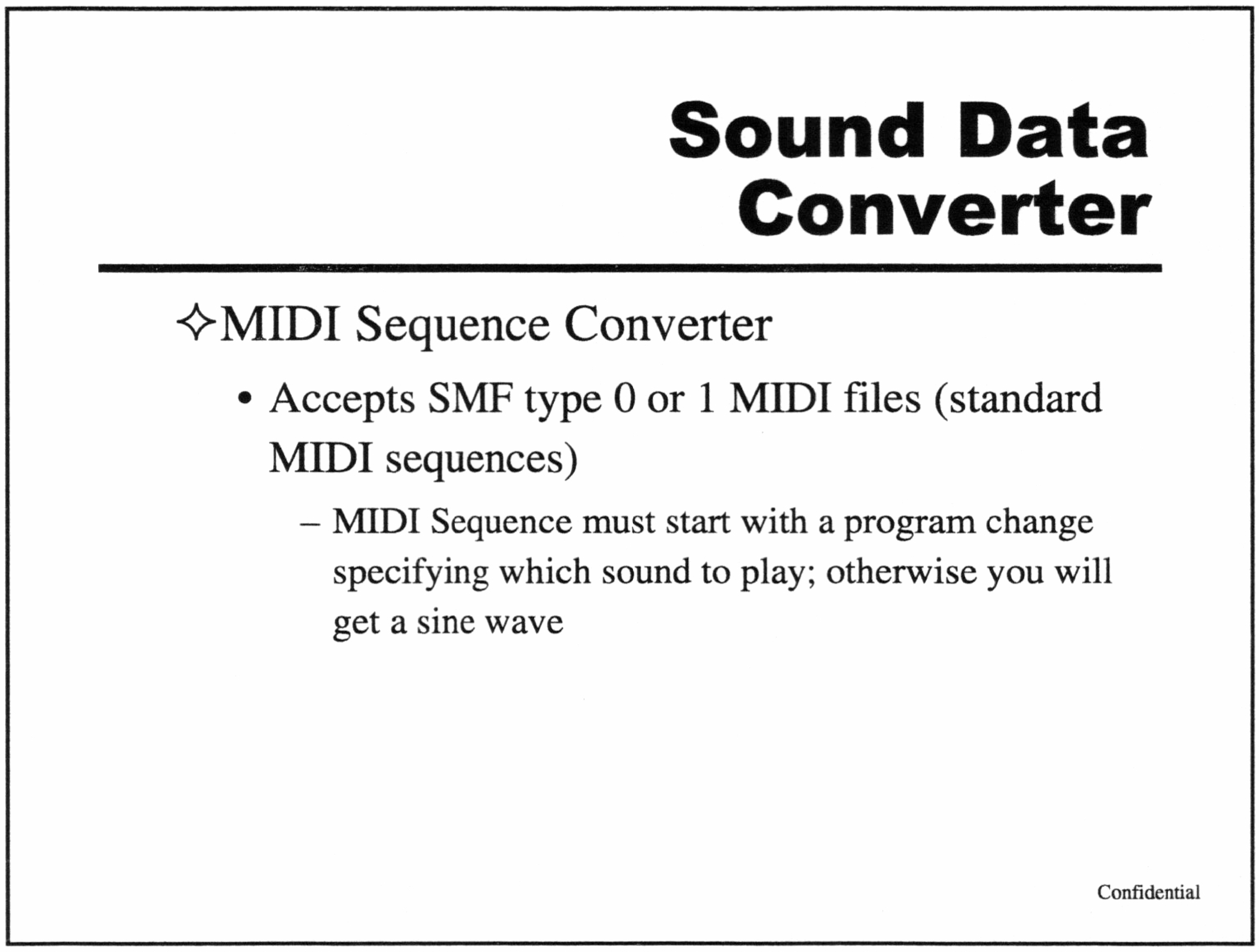


Sound Data Converter

✧One-Shot data converter (continued)

- Demo

Confidential



Sound Data Converter

✧MIDI Sequence Converter

- Accepts SMF type 0 or 1 MIDI files (standard MIDI sequences)
  - MIDI Sequence must start with a program change specifying which sound to play; otherwise you will get a sine wave

Confidential

Sound Data Converter

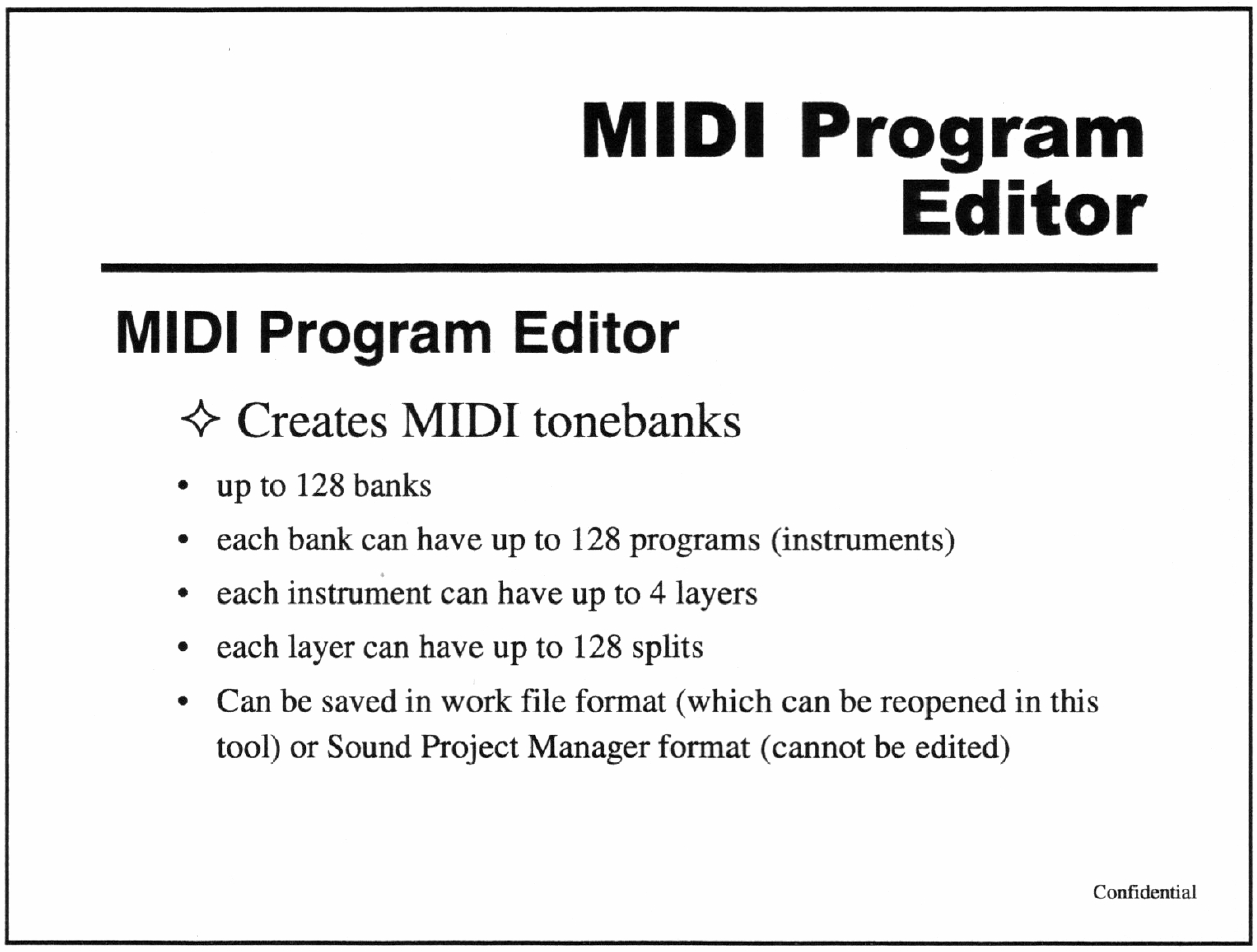
✧MIDI Sequence Converter (continued)

- Size/number of sequences is limited only by sound memory
  - 48 note polyphony maximum, anything over that will not sound, and there is a possibility that this will result in audio errors (stuck notes, etc.)
- Output file is .msb used in Sound Project Manager

Sound Data Converter

✧MIDI Sequence Converter (continued)

- Demo



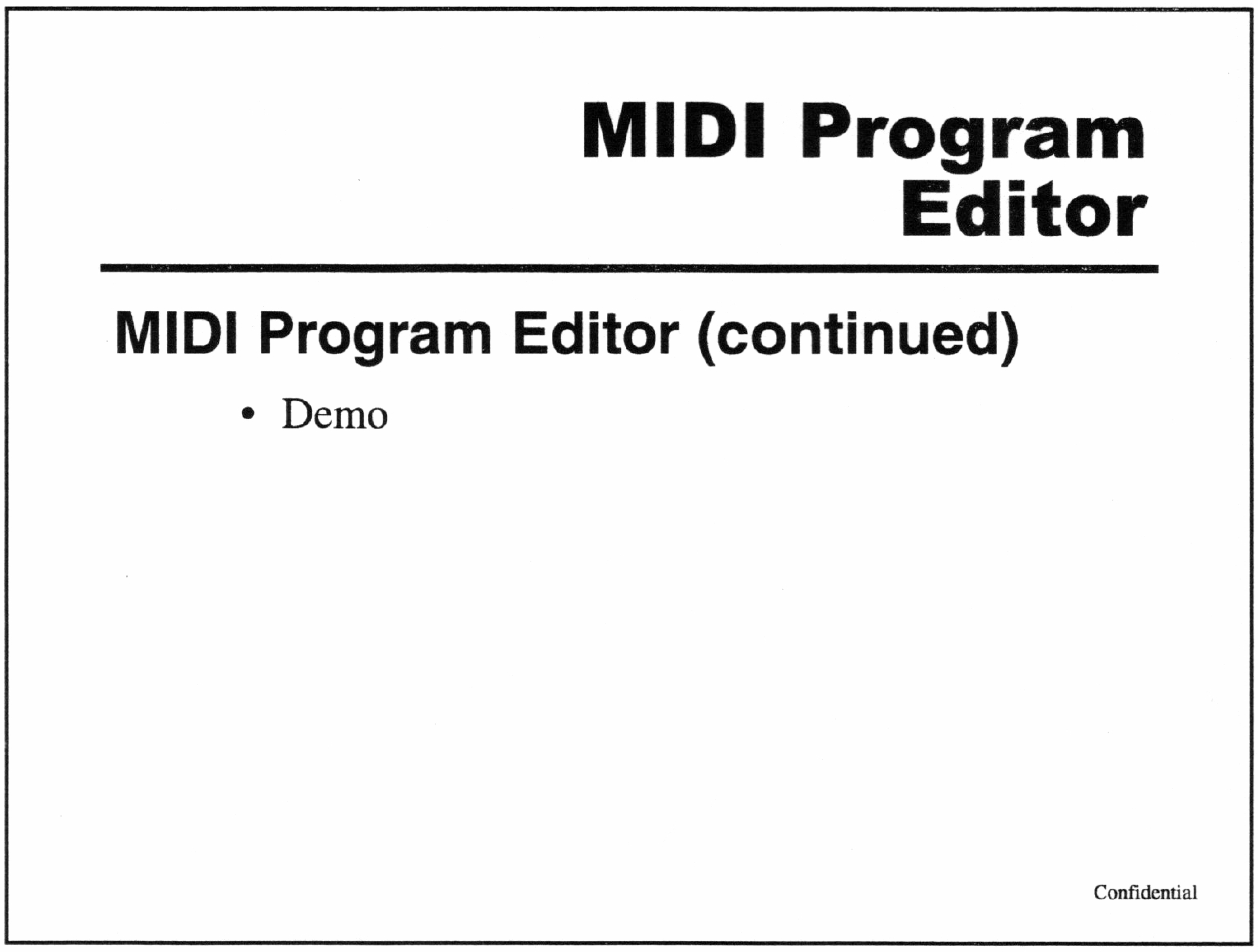
MIDI Program Editor

MIDI Program Editor

✧ Creates MIDI tonebanks

- up to 128 banks
- each bank can have up to 128 programs (instruments)
- each instrument can have up to 4 layers
- each layer can have up to 128 splits
- Can be saved in work file format (which can be reopened in this tool) or Sound Project Manager format (cannot be edited)

Confidential

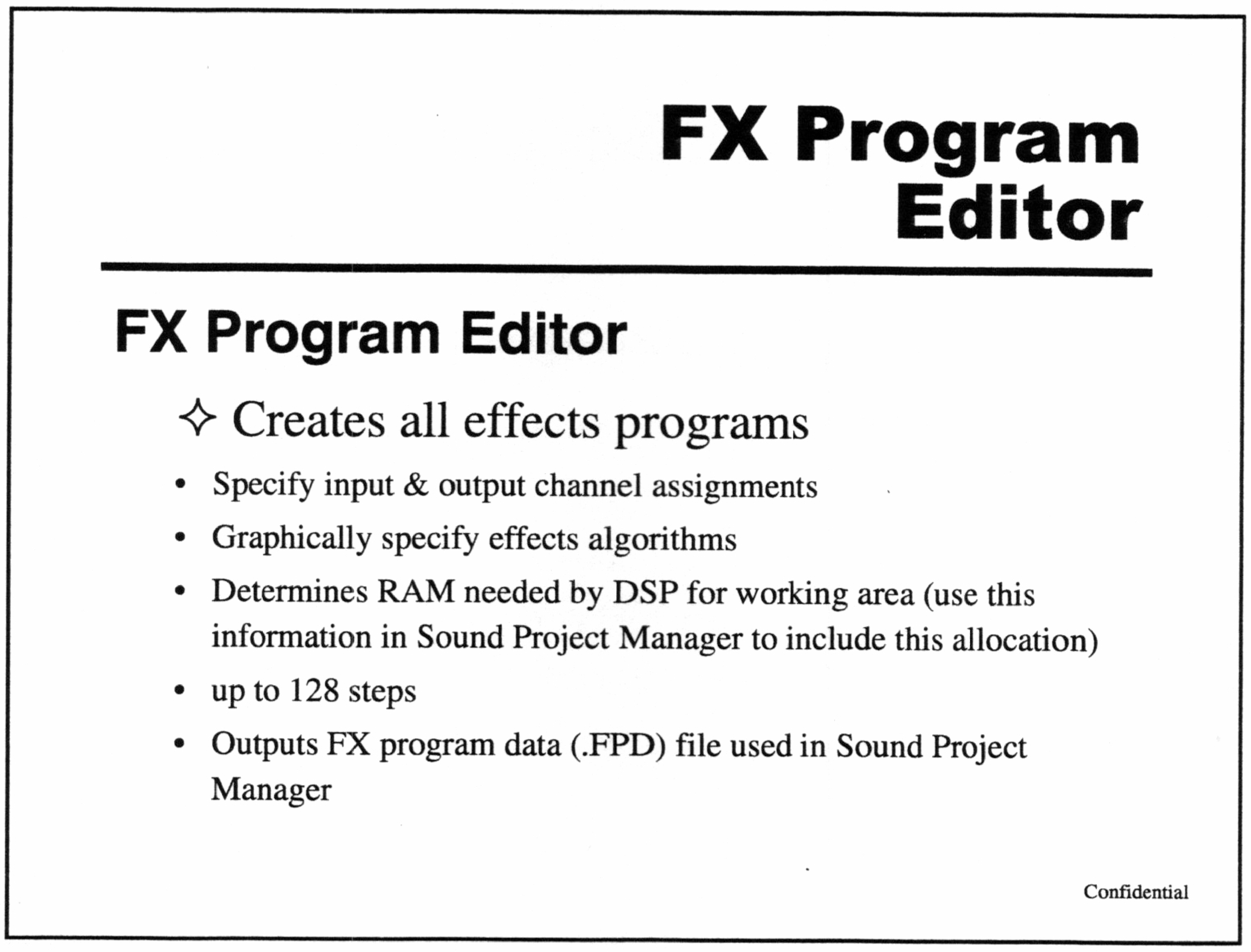


MIDI Program Editor

MIDI Program Editor (continued)

- Demo

Confidential

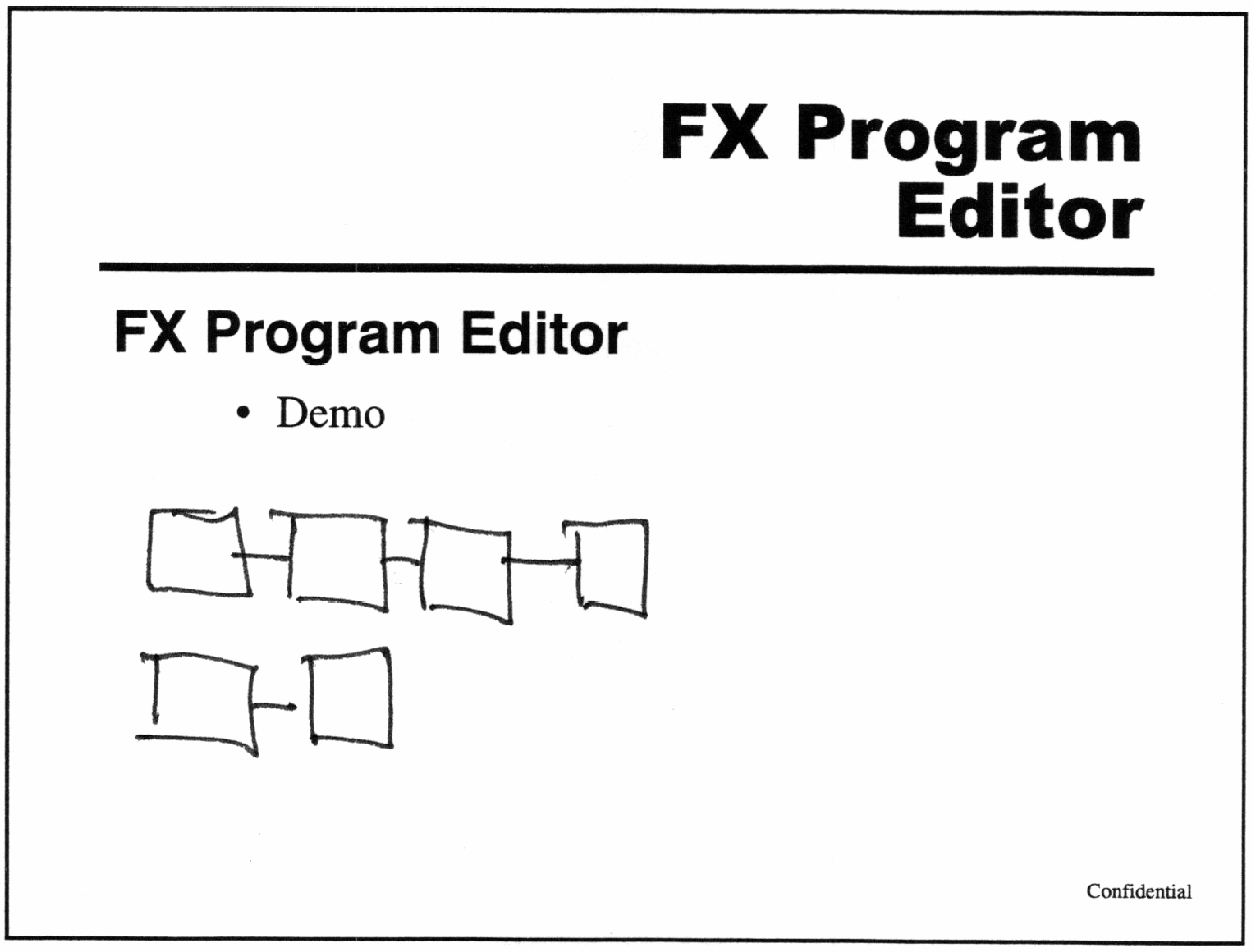


FX Program Editor

FX Program Editor

✧ Creates all effects programs

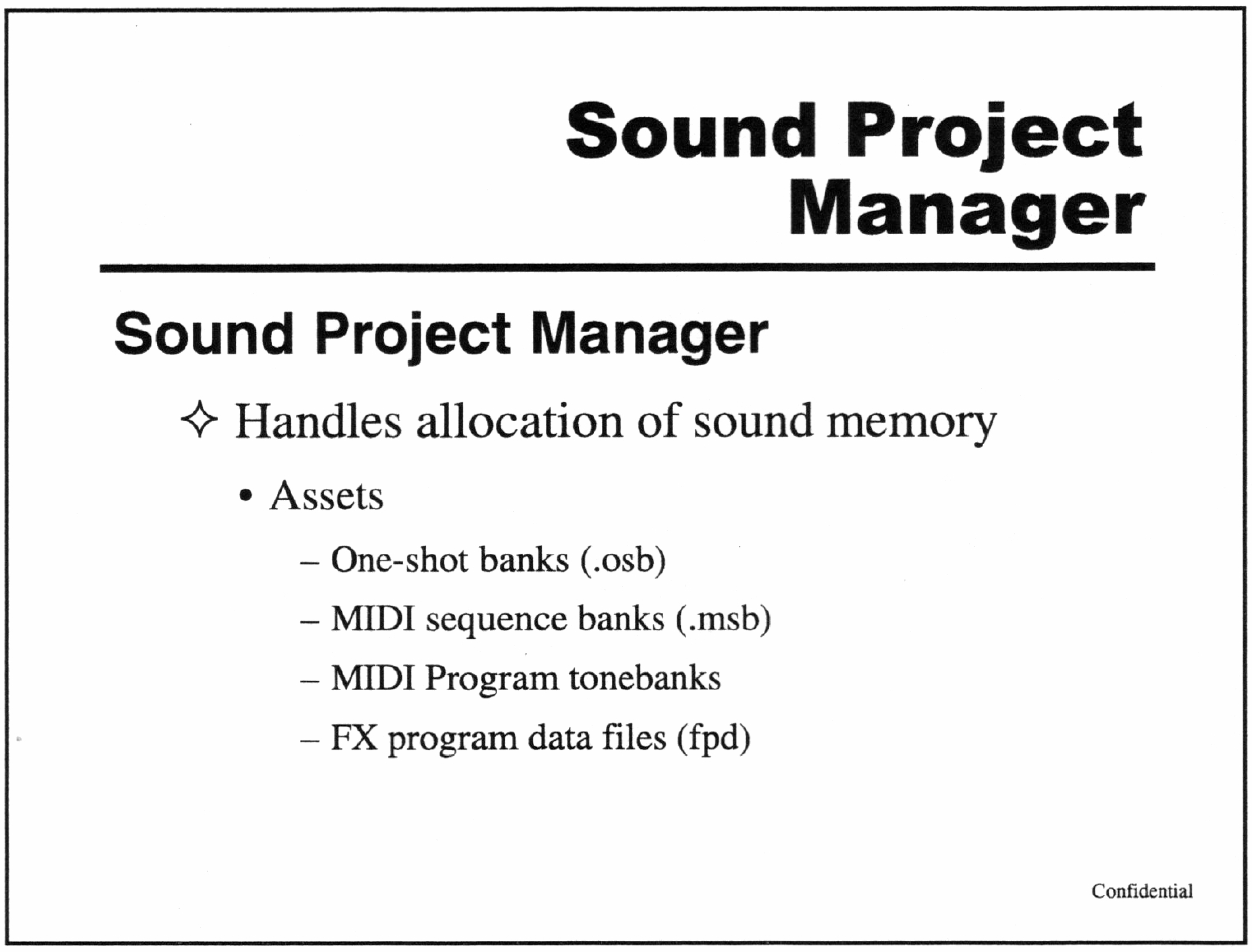
- Specify input & output channel assignments
- Graphically specify effects algorithms
- Determines RAM needed by DSP for working area (use this information in Sound Project Manager to include this allocation)
- up to 128 steps
- Outputs FX program data (.FPD) file used in Sound Project Manager



FX Program Editor

FX Program Editor

- Demo

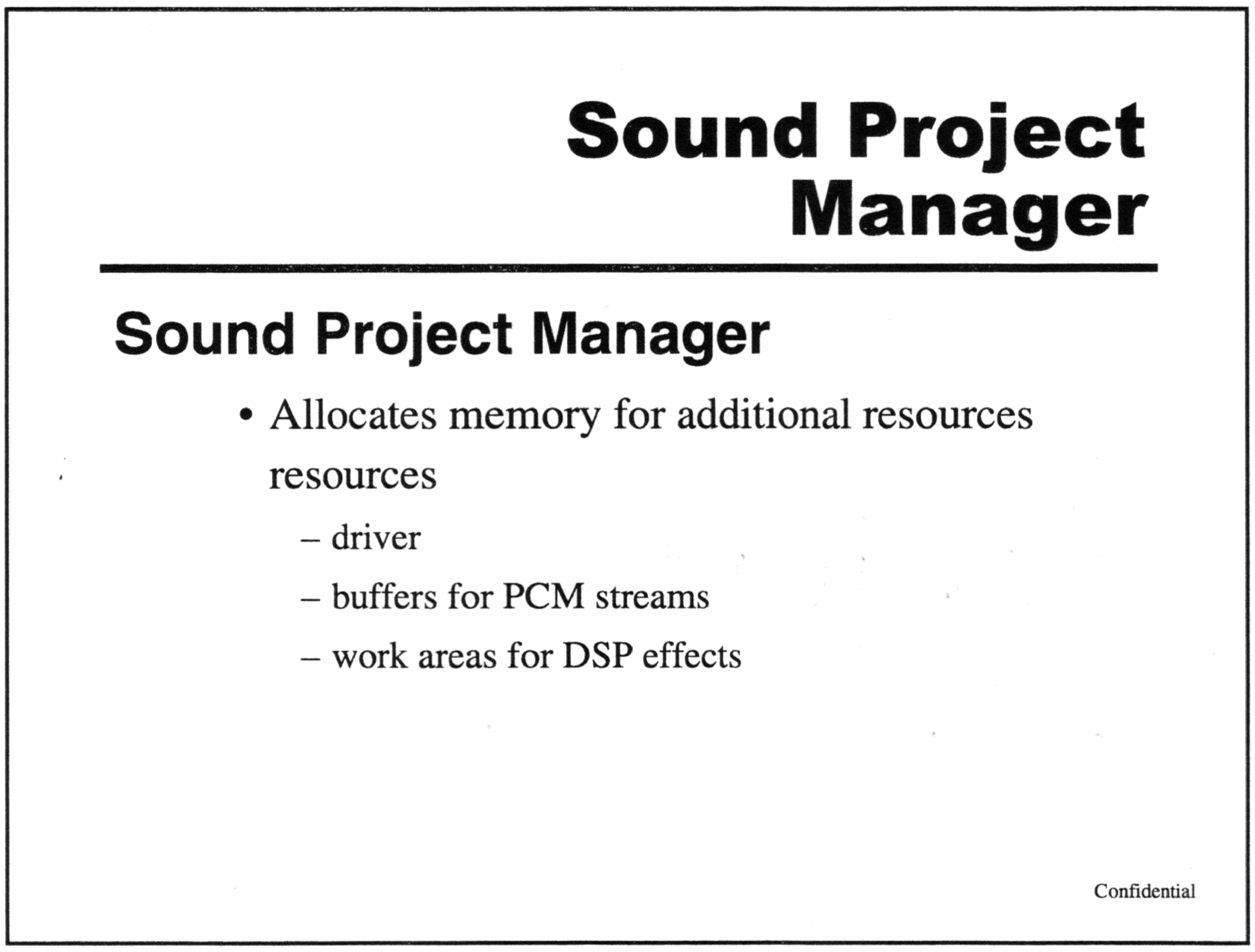


Sound Project Manager

Sound Project Manager

- ✧ Handles allocation of sound memory
  - Assets
    - – One-shot banks (.osb)
    - – MIDI sequence banks (.msb)
    - – MIDI Program tonebanks
    - – FX program data files (fpd)

Confidential



Sound Project Manager

Sound Project Manager

- Allocates memory for additional resources resources
  - – driver
  - – buffers for PCM streams
  - – work areas for DSP effects

Confidential

Sound Project Manager

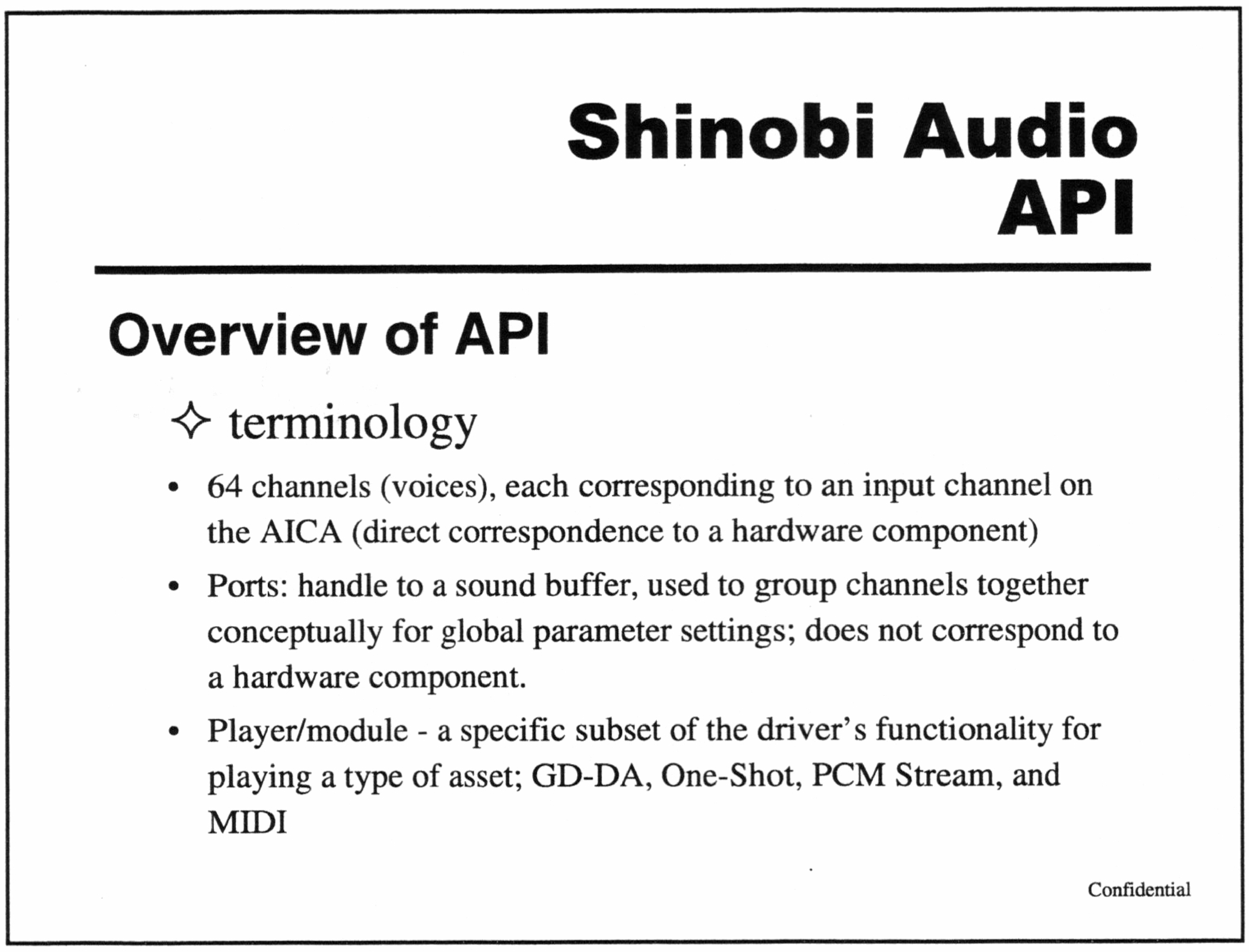
Sound Project Manager

- Outputs MultiUnit files (.mlt) Final output
  - contains allocation information for each type of resource
  - contains sound and program data (all assets except PCM stream and GD-DA)
  - Needed for every asset that requires sound memory (eg., all except GD-DA)

Sound Project Manager

Sound Project Manager

- Demo



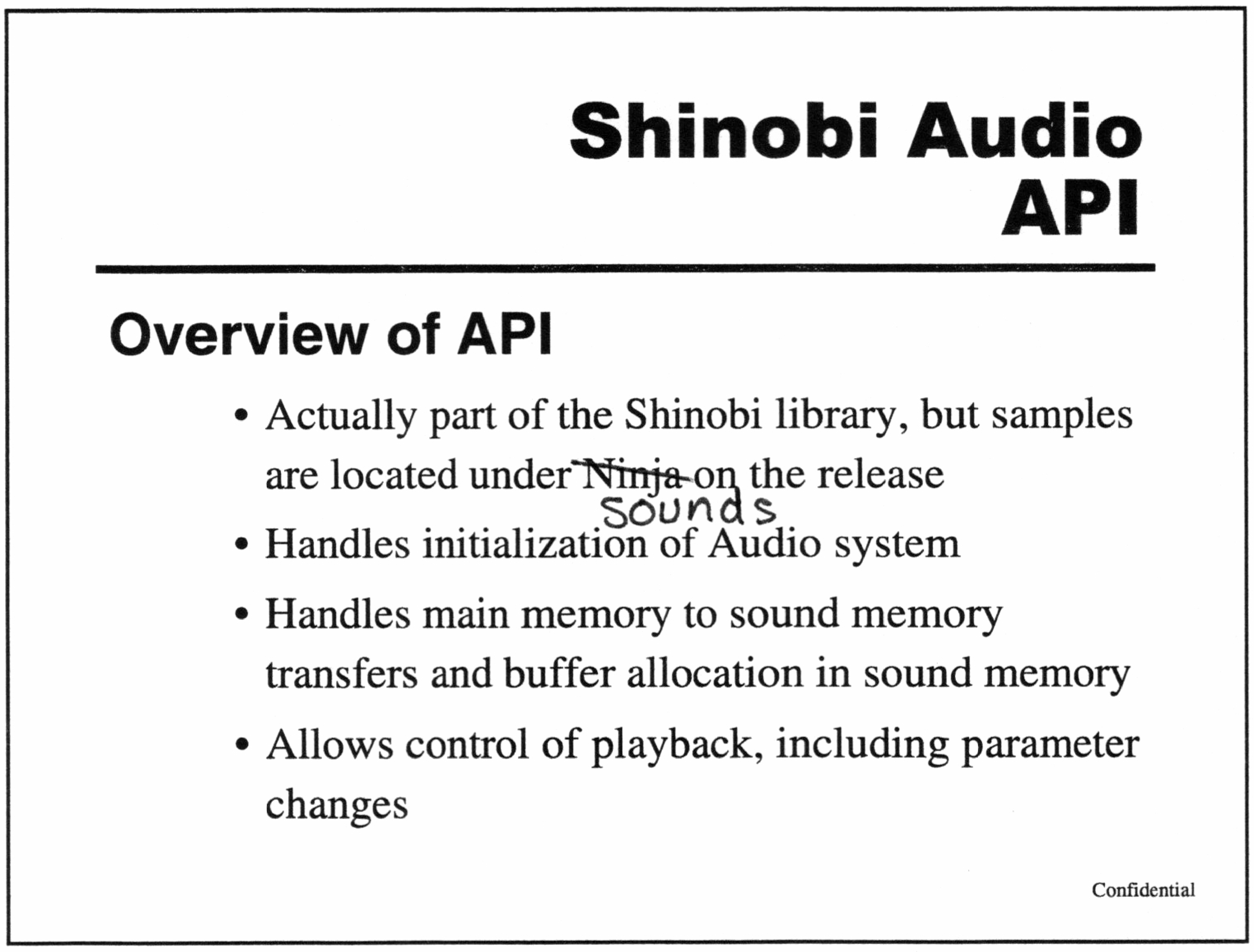
Shinobi Audio API

Overview of API

✧ terminology

- 64 channels (voices), each corresponding to an input channel on the AICA (direct correspondence to a hardware component)
- Ports: handle to a sound buffer, used to group channels together conceptually for global parameter settings; does not correspond to a hardware component.
- Player/module - a specific subset of the driver's functionality for playing a type of asset; GD-DA, One-Shot, PCM Stream, and MIDI

Confidential

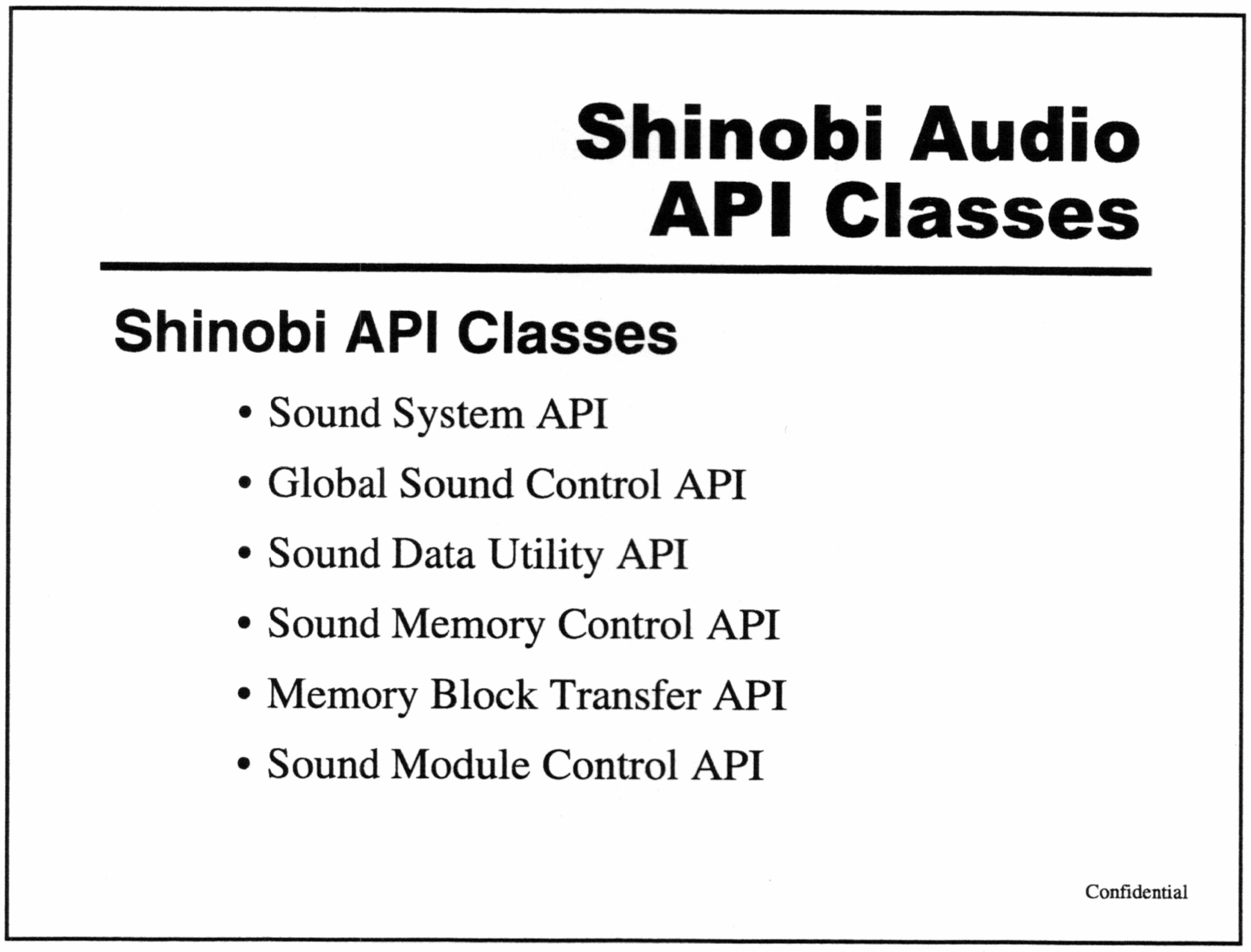


Shinobi Audio API

Overview of API

- Actually part of the Shinobi library, but samples are located under ~~Ninja~~ sounds on the release
- Handles initialization of Audio system
- Handles main memory to sound memory transfers and buffer allocation in sound memory
- Allows control of playback, including parameter changes

Confidential

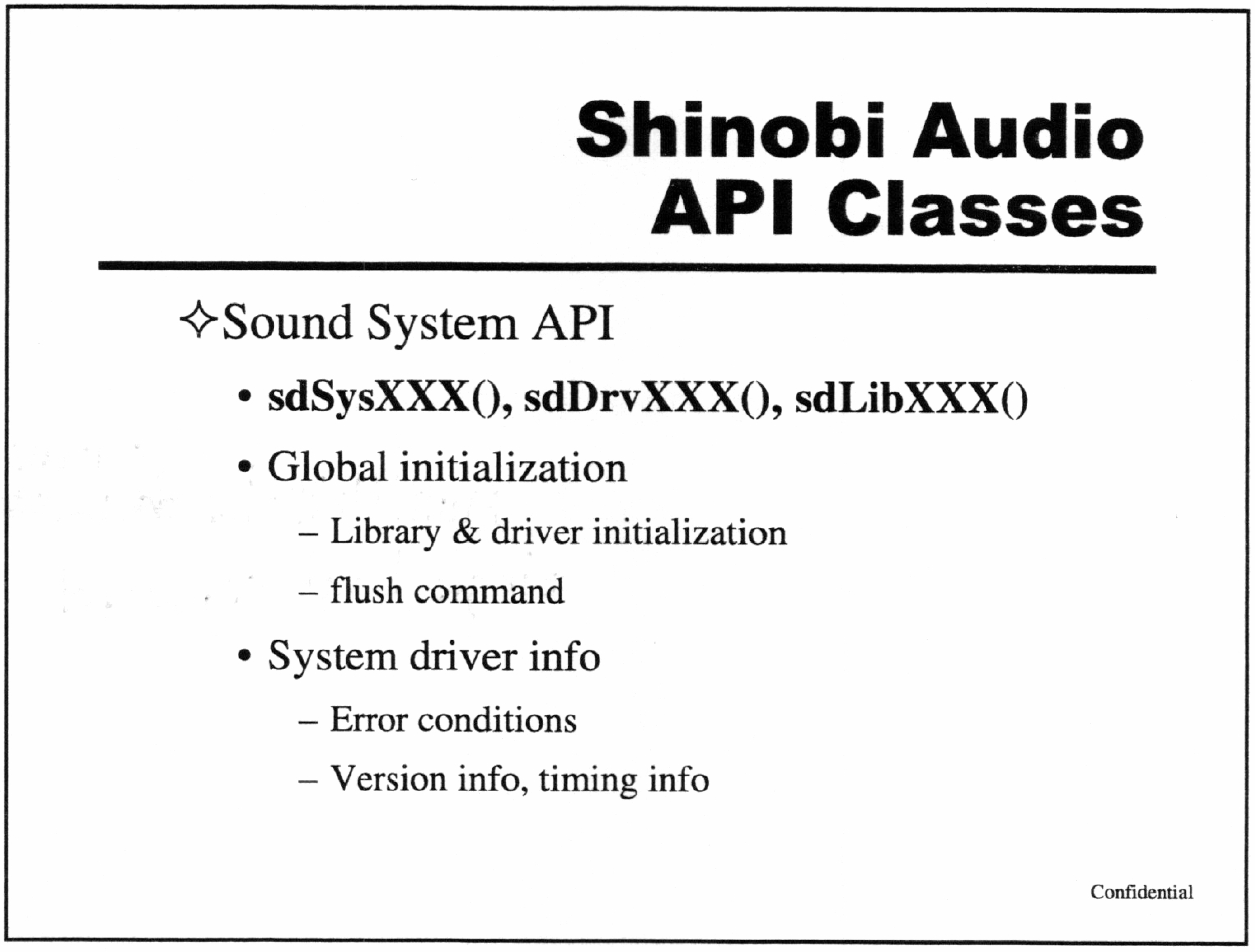


Shinobi Audio API Classes

Shinobi API Classes

- Sound System API
- Global Sound Control API
- Sound Data Utility API
- Sound Memory Control API
- Memory Block Transfer API
- Sound Module Control API

Confidential



Shinobi Audio API Classes

✧Sound System API

- **sdSysXXX(), sdDrvXXX(), sdLibXXX()**
- Global initialization
  - Library & driver initialization
  - flush command
- System driver info
  - Error conditions
  - Version info, timing info

Confidential

Shinobi Audio API Classes

✧Global Sound Control API

- **sdSndXXX()**
- Total volume, stereo/mono toggle
- DSP control
  - – stop command
  - – set active channels and programs
  - – Qsound positioning

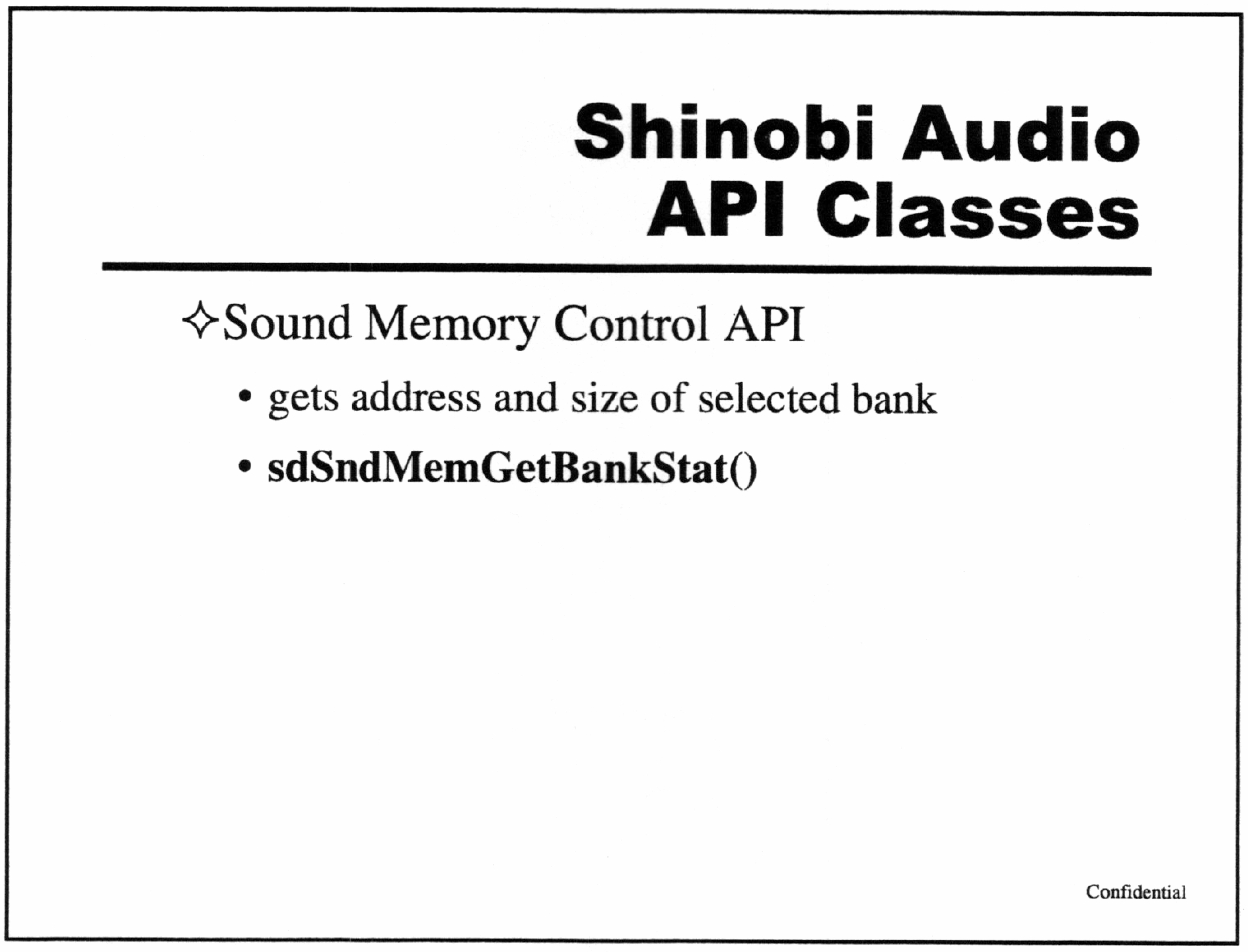
Confidential

Shinobi Audio API Classes

✧Sound Data Utility API

- Transfer MultiUnit or bank from main memory to sound memory
  - – **sdMultiUnitDownload()** MultiUnit file ~~downloaded~~ transfered to sound memory
  - – **sdBankDownload()** single bank replaced

Confidential

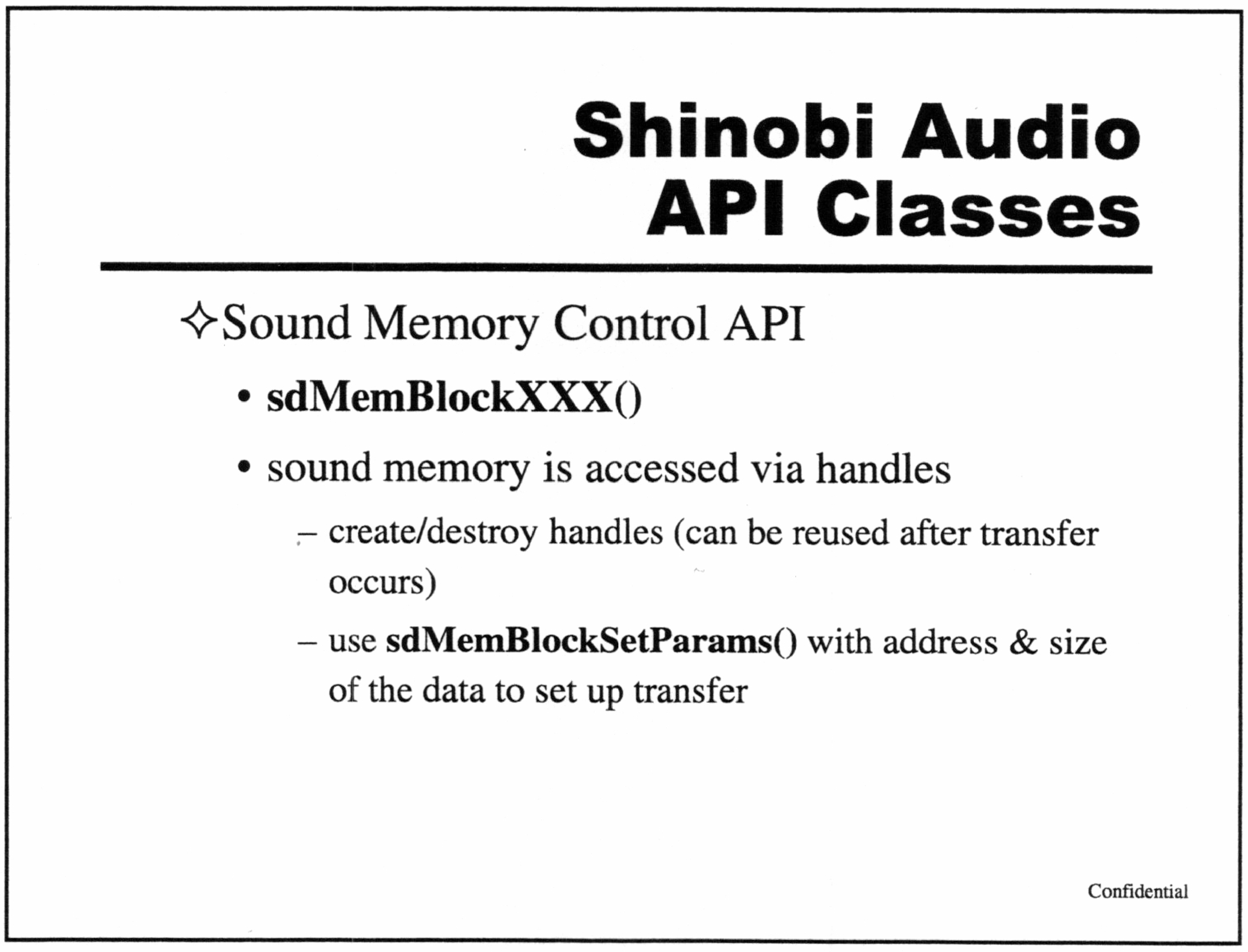


Shinobi Audio API Classes

✧Sound Memory Control API

- gets address and size of selected bank
- **sdSndMemGetBankStat()**

Confidential

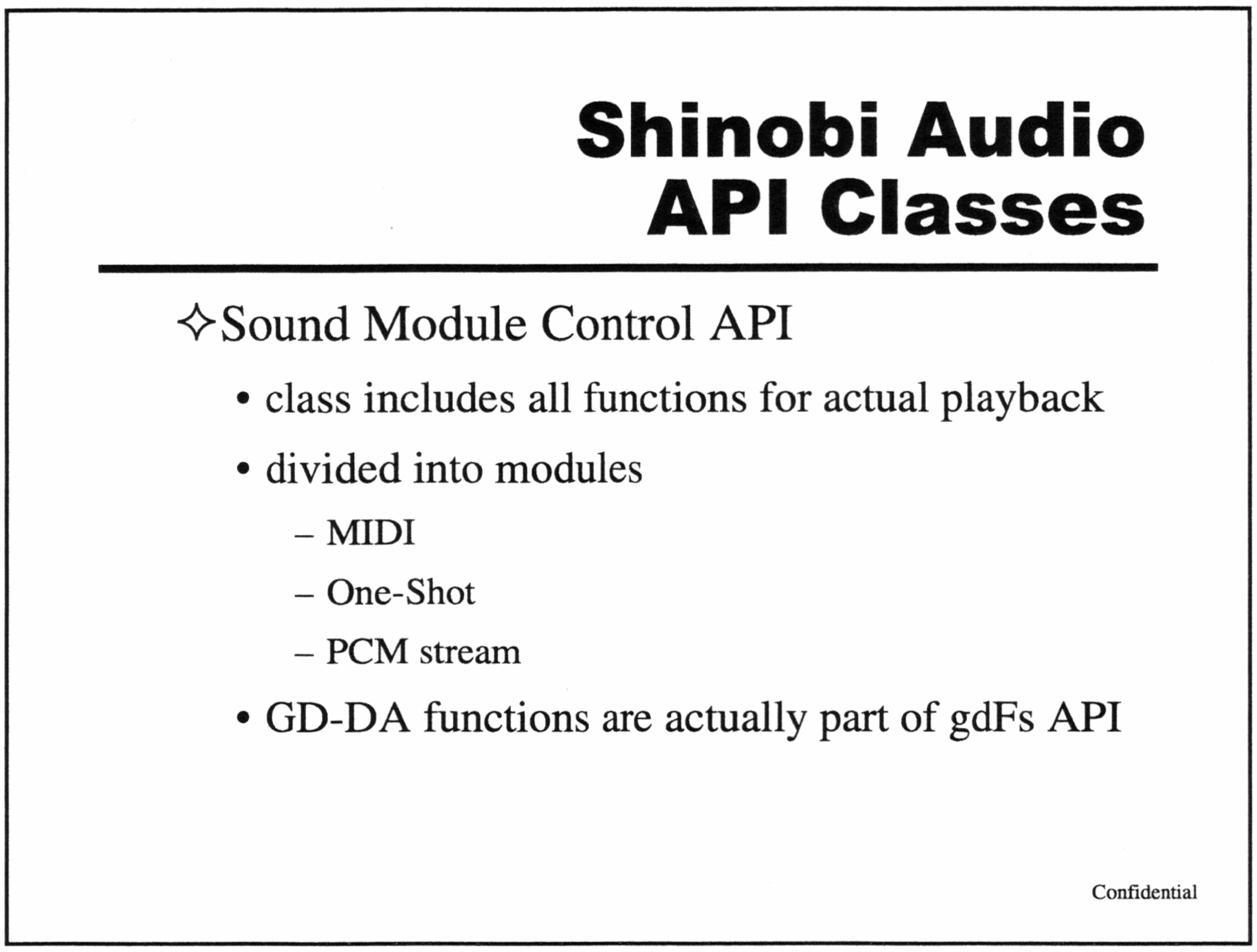


Shinobi Audio API Classes

✧Sound Memory Control API

- **sdMemBlockXXX()**
- sound memory is accessed via handles
  - create/destroy handles (can be reused after transfer occurs)
  - use **sdMemBlockSetParams()** with address & size of the data to set up transfer

Confidential

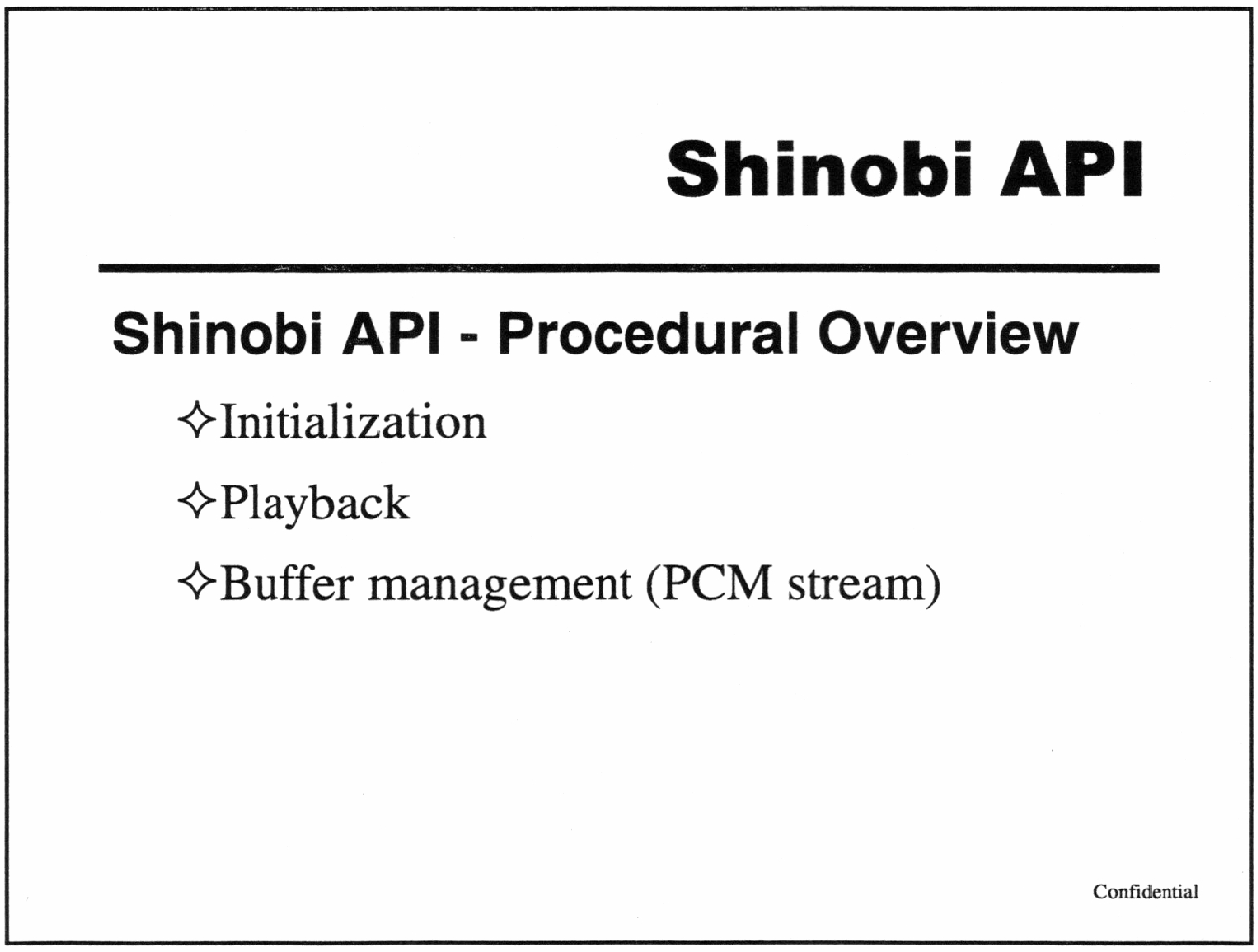


Shinobi Audio API Classes

✧Sound Module Control API

- class includes all functions for actual playback
- divided into modules
  - MIDI
  - One-Shot
  - PCM stream
- GD-DA functions are actually part of gdFs API

Confidential



Shinobi API

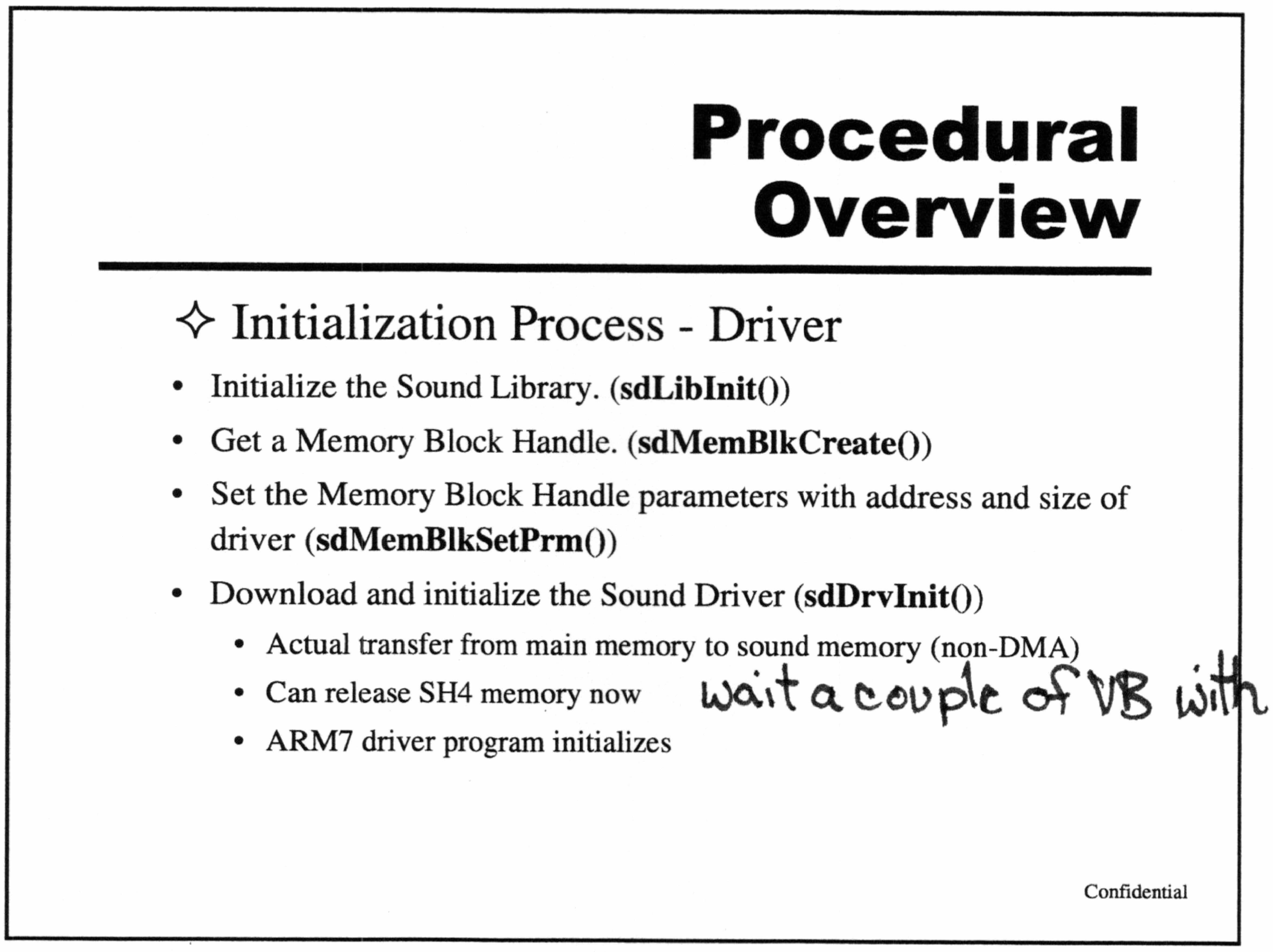
Shinobi API - Procedural Overview

✧Initialization

✧Playback

✧Buffer management (PCM stream)

Confidential

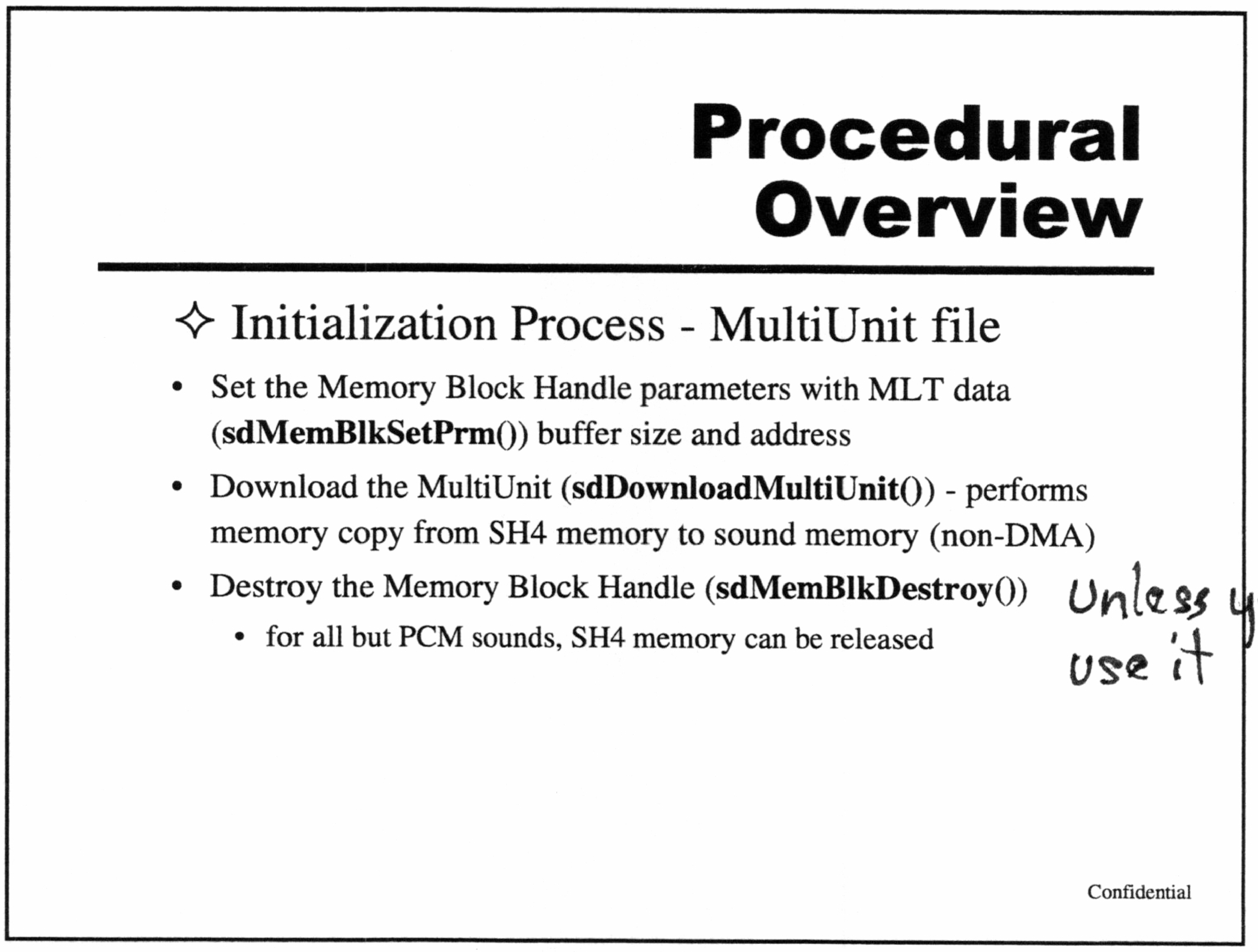


Procedural Overview

✧ Initialization Process - Driver

- Initialize the Sound Library. (**sdLibInit**())
- Get a Memory Block Handle. (**sdMemBlkCreate()**)
- Set the Memory Block Handle parameters with address and size of driver (**sdMemBlkSetPrm()**)
- Download and initialize the Sound Driver (**sdDrvInit**())
  - Actual transfer from main memory to sound memory (non-DMA)
  - Can release SH4 memory now wait a couple of VB with old driver
  - ARM7 driver program initializes

Confidential



Procedural Overview

✧ Initialization Process - MultiUnit file

- Set the Memory Block Handle parameters with MLT data (**sdMemBlkSetPrm**()) buffer size and address
- Download the MultiUnit (**sdDownloadMultiUnit()**) - performs memory copy from SH4 memory to sound memory (non-DMA)
- Destroy the Memory Block Handle (**sdMemBlkDestroy**()) Unless you're going to use it later.
  - for all but PCM sounds, SH4 memory can be released

Confidential

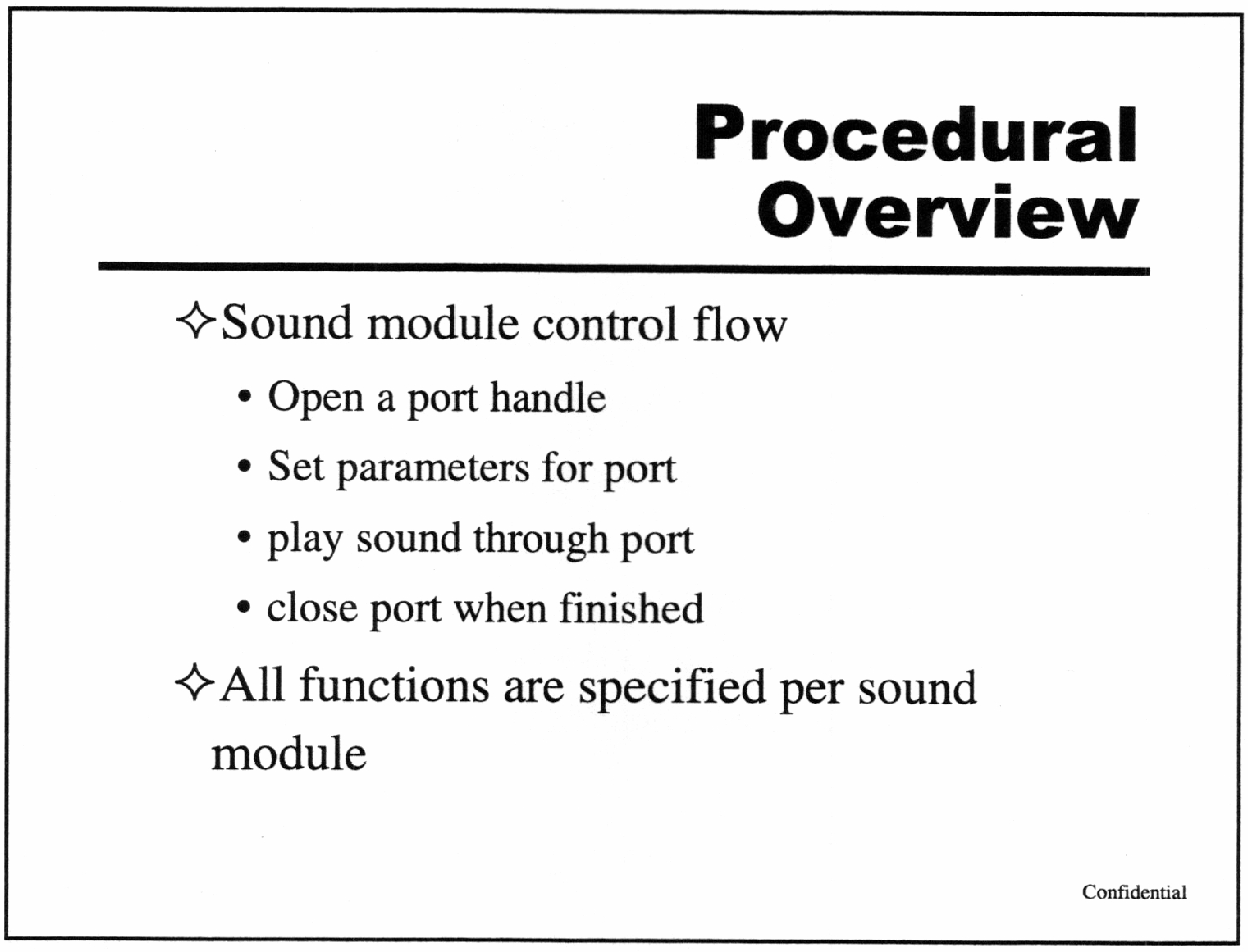
Procedural Overview

- Example (SoundInit.c)

Procedural Overview

✧Audio Playback

- Control Flow
- Module Specifications
- Buffer Management



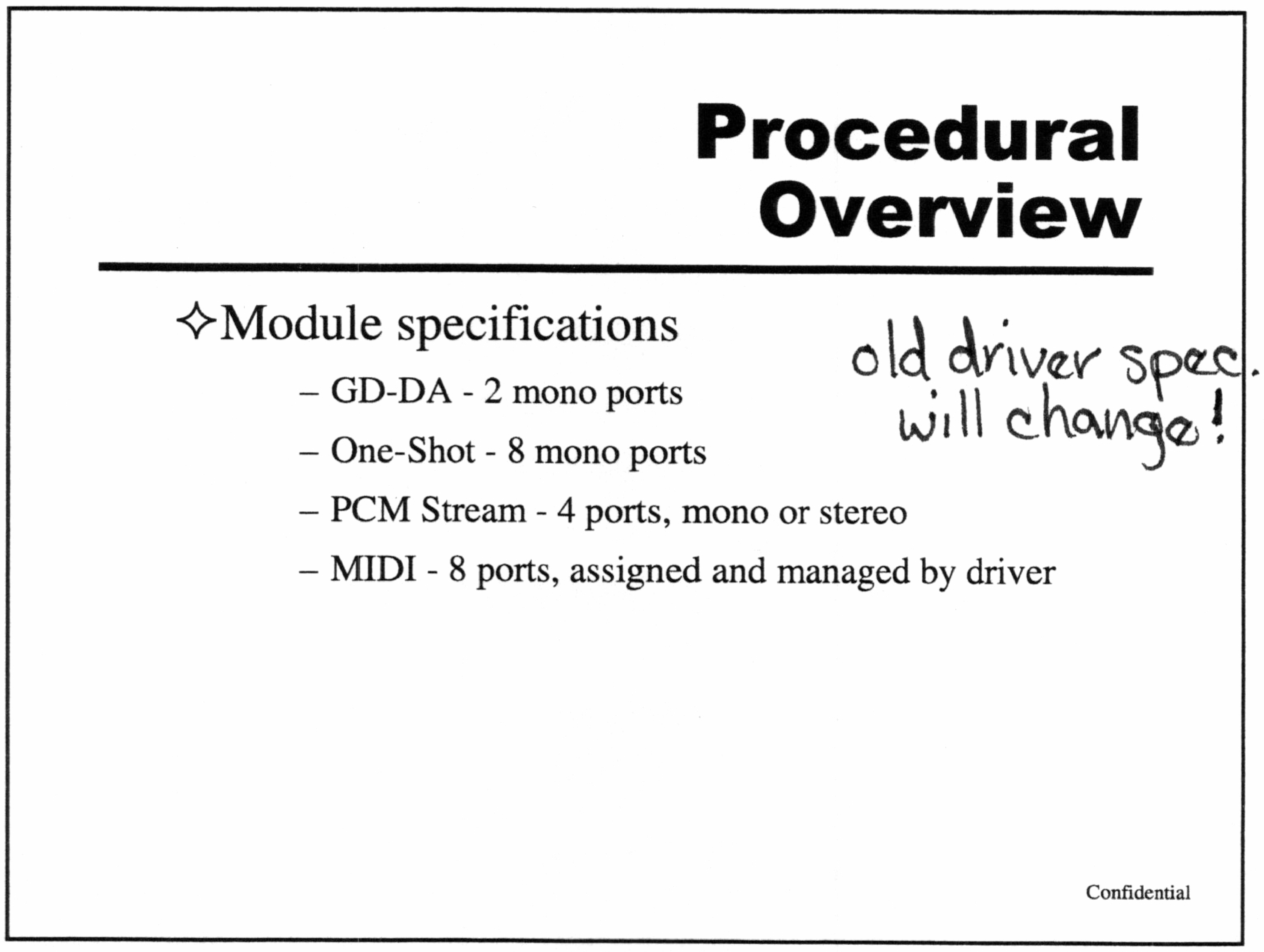
Procedural Overview

✧Sound module control flow

- Open a port handle
- Set parameters for port
- play sound through port
- close port when finished

✧All functions are specified per sound module

Confidential



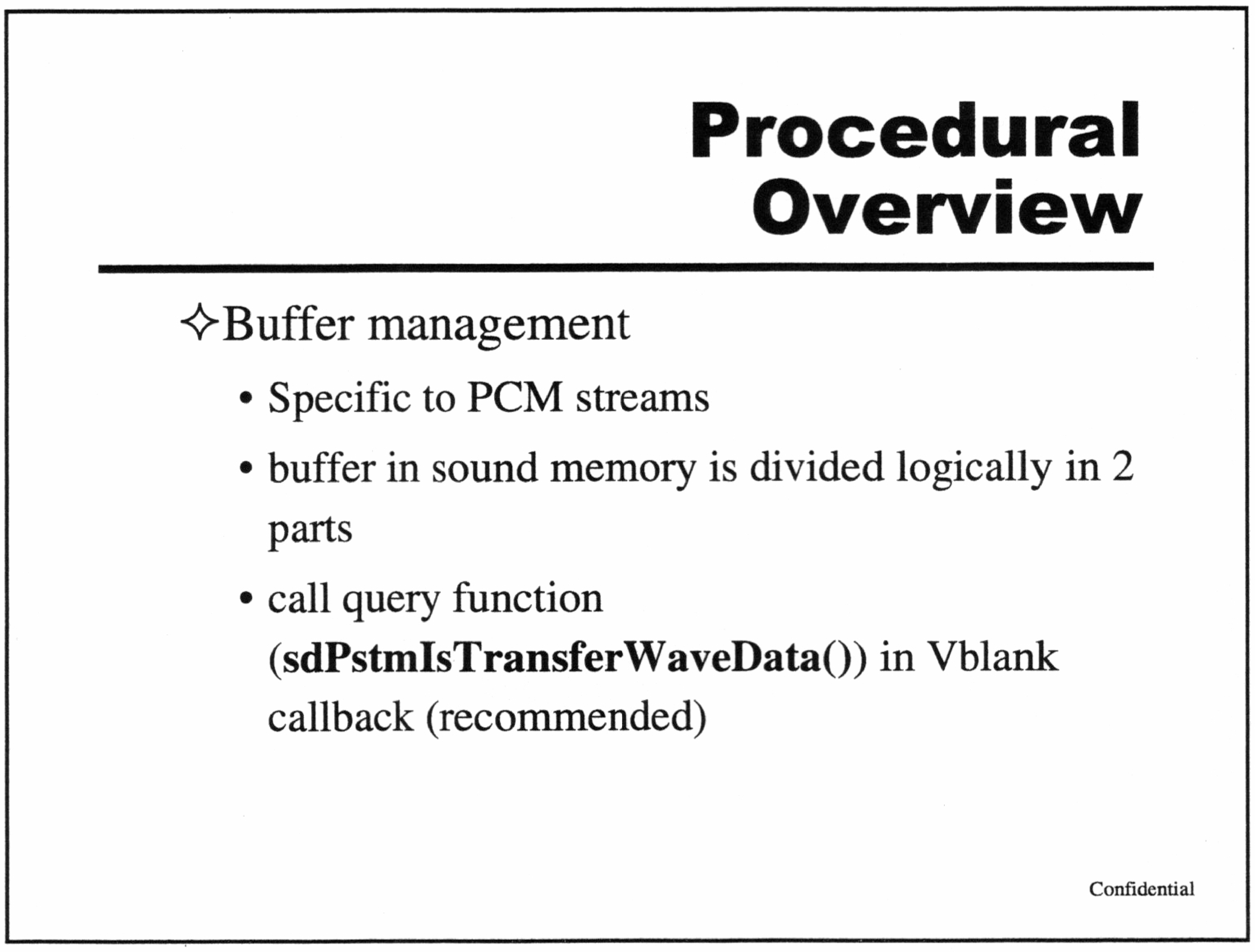
Procedural Overview

✧Module specifications

- – GD-DA - 2 mono ports
- – One-Shot - 8 mono ports
- – PCM Stream - 4 ports, mono or stereo
- – MIDI - 8 ports, assigned and managed by driver

old driver spec.
will change!

Confidential

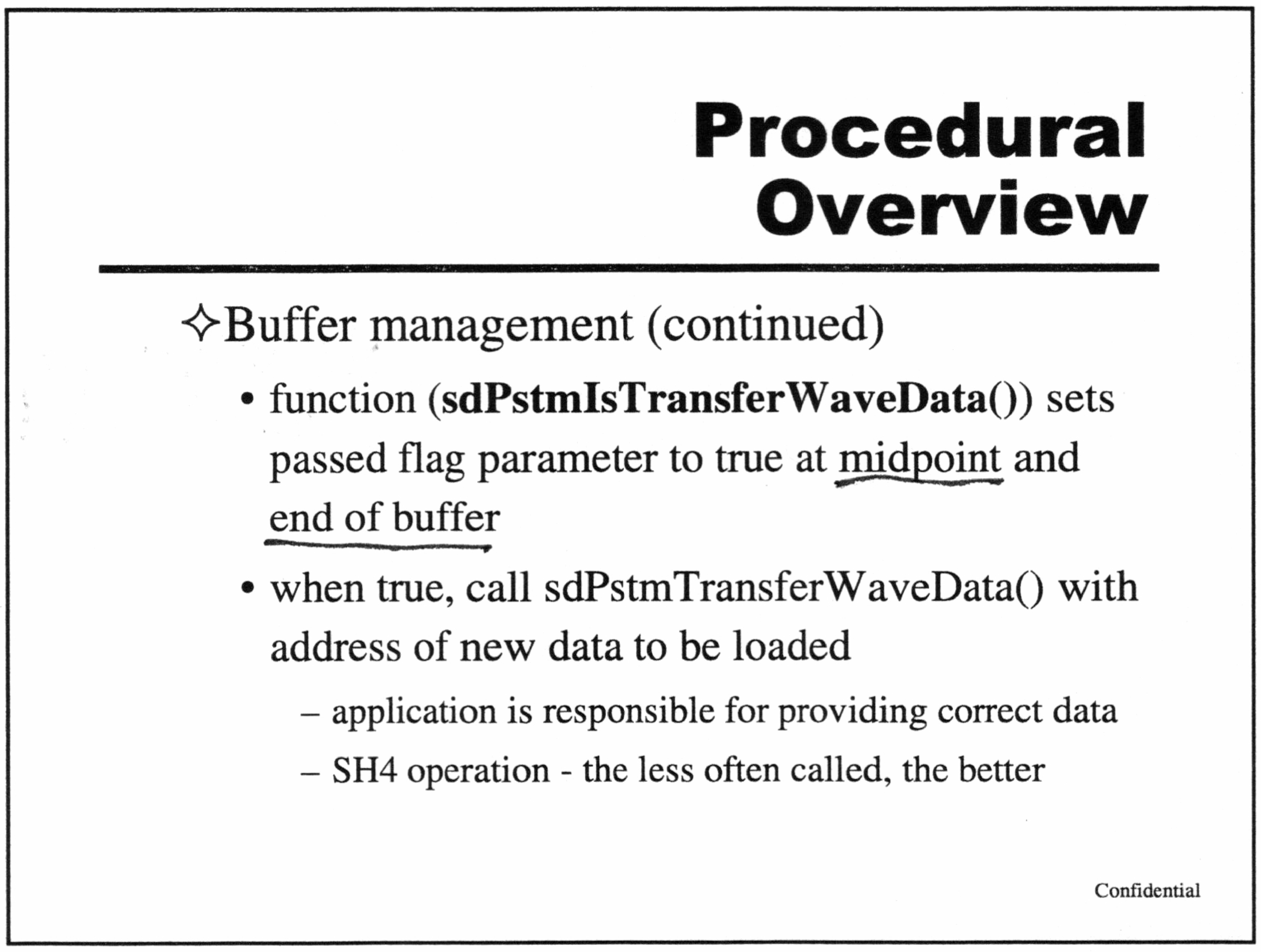


Procedural Overview

✧Buffer management

- Specific to PCM streams
- buffer in sound memory is divided logically in 2 parts
- call query function (**sdPstmIsTransferWaveData()**) in Vblank callback (recommended)

Confidential

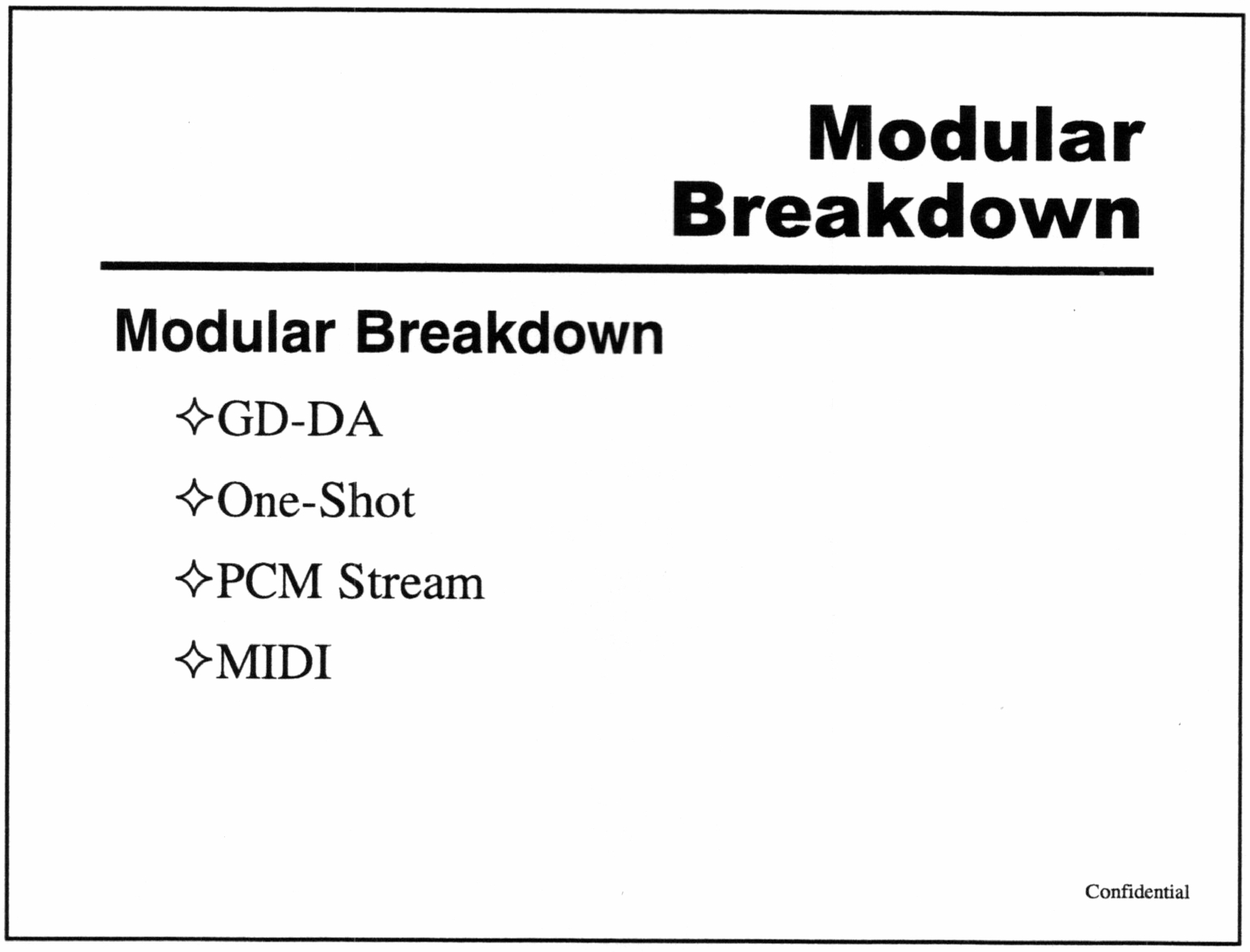


Procedural Overview

✧Buffer management (continued)

- function (**sdPstmIsTransferWaveData()**) sets passed flag parameter to true at midpoint and end of buffer
- when true, call sdPstmTransferWaveData() with address of new data to be loaded
  - application is responsible for providing correct data
  - SH4 operation - the less often called, the better

Confidential

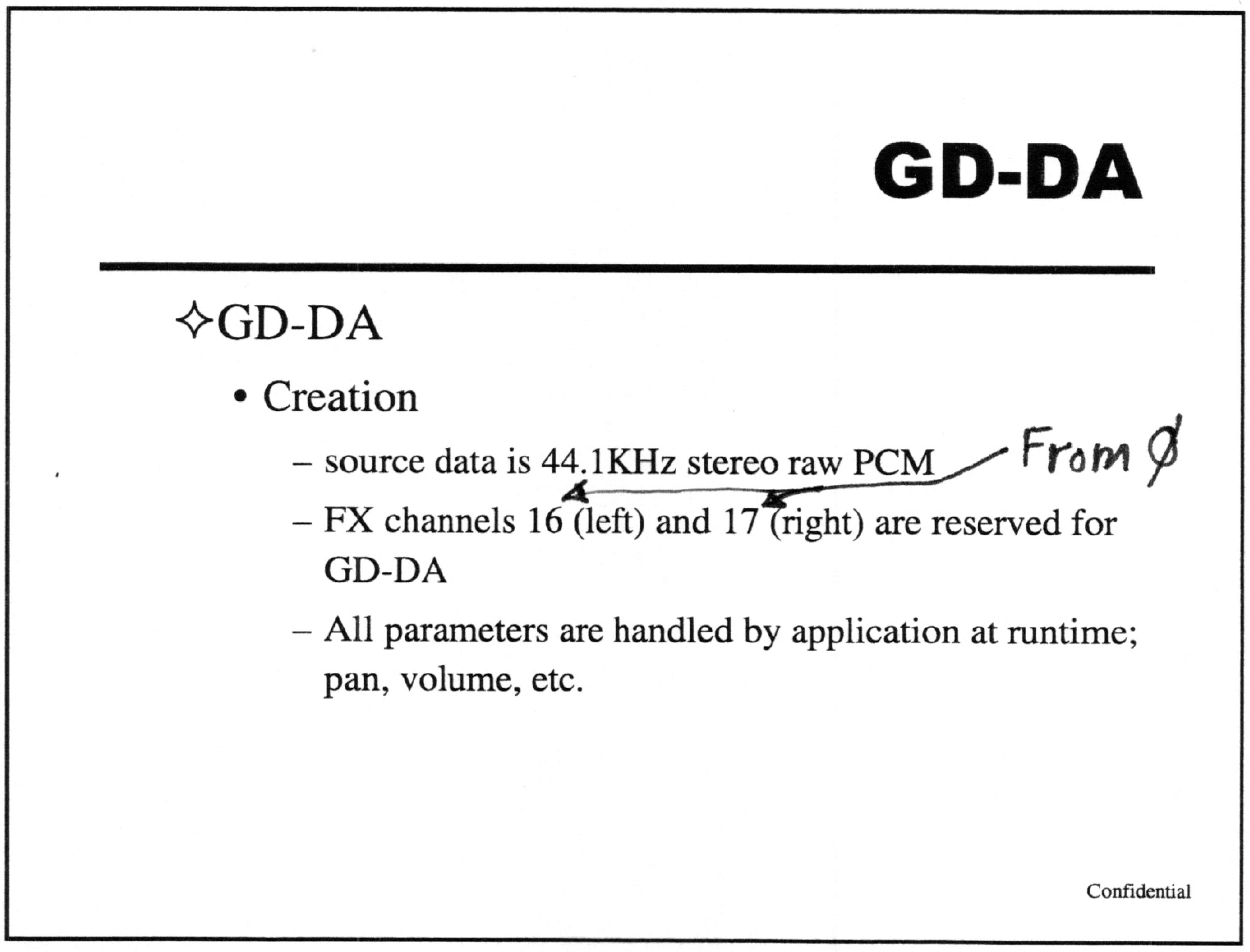


Modular Breakdown

Modular Breakdown

- ✧GD-DA
- ✧One-Shot
- ✧PCM Stream
- ✧MIDI

Confidential



GD-DA

✧GD-DA

- Creation
  - source data is 44.1KHz stereo raw PCM From Ø
  - FX channels 16 (left) and 17 (right) are reserved for GD-DA
  - All parameters are handled by application at runtime; pan, volume, etc.

Confidential

GD-DA

✧GD-DA

- Parameter settings
  - volume and pan are set individually for left and right
  - speed/pitch is set for both as a stereo port
  - Any FX program on channels 16 & 17 will be applied to left and right channels, respectively

GD-DA

- Playback
  - Start (pass track # as parameter), stop, play sector, pause, release (unpause)
  - Can play a span of sequential tracks

GD-DA

- Performance considerations
  - Small system load, no memory used
  - Trade-off between amount of sound data on CD vs. program data.

GD-DA

- General considerations
  - No other GD access is possible while GD-DA is playing
  - can be treated as a single stereo stream or as 2 separate mono streams, each with its own pan, volume, and FX.
  - Short pause before playing begins (not a problem for background music, but too long for sound effect usage), < 200 ms (GD-ROM seek time)

GD-DA

- Example (GD-DA.c)

SOS loads a dummy Multi Unit file

Confidential

One-Shot

✧One-Shot Bank

- Creation
  - monaural 4-bit, 8-bit, or 16-bit, 11.025KHz, 22.05KHz, or 44.1KHz sounds, < 64k frames each
  - Demo - Sound Data Converter -> Sound Project Manager

Confidential

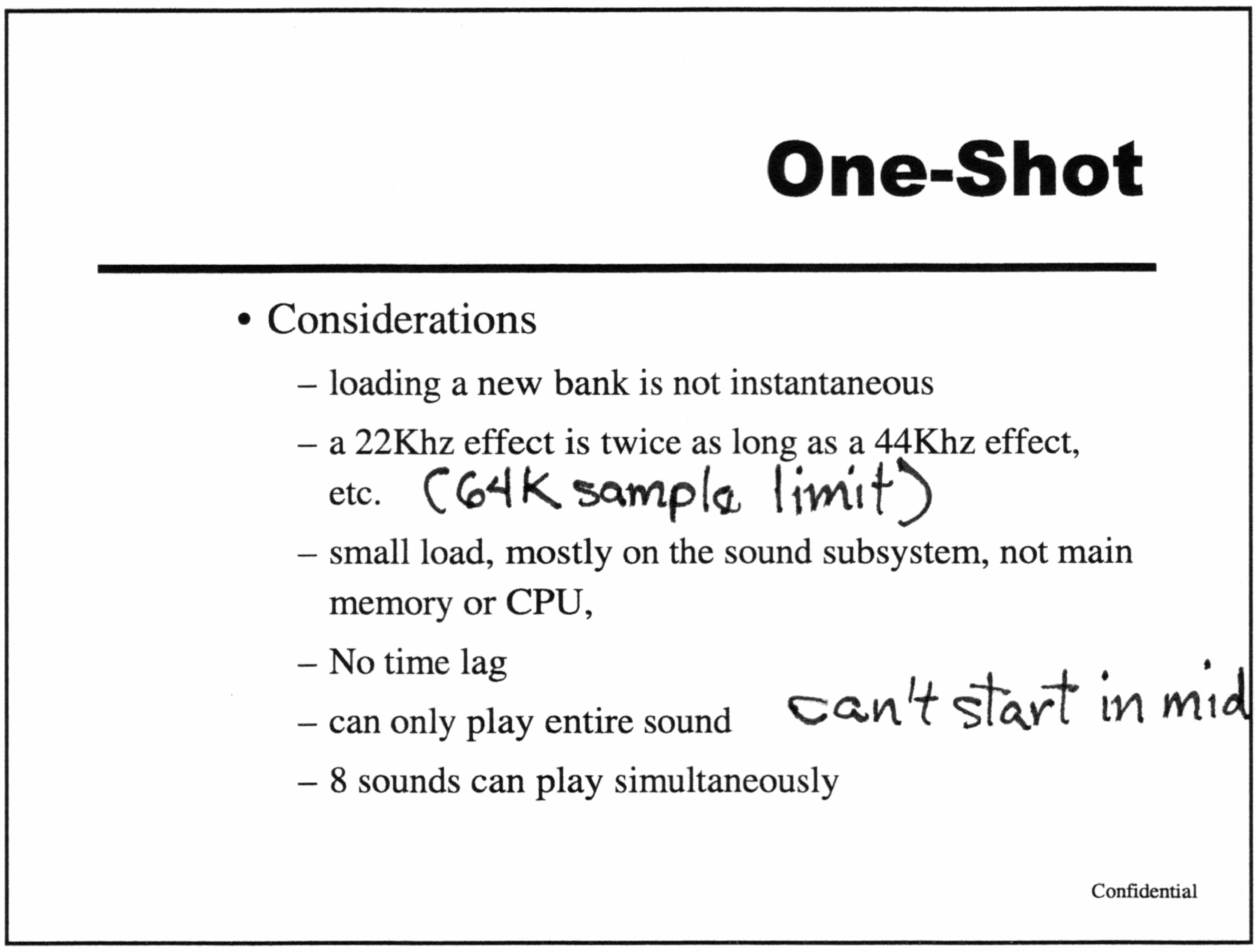
One-Shot

✧One-Shot Bank

- Initialization
  - – is part of MultiUnit Initialization
  - – can swap in new banks without reinitializing driver

One-Shot

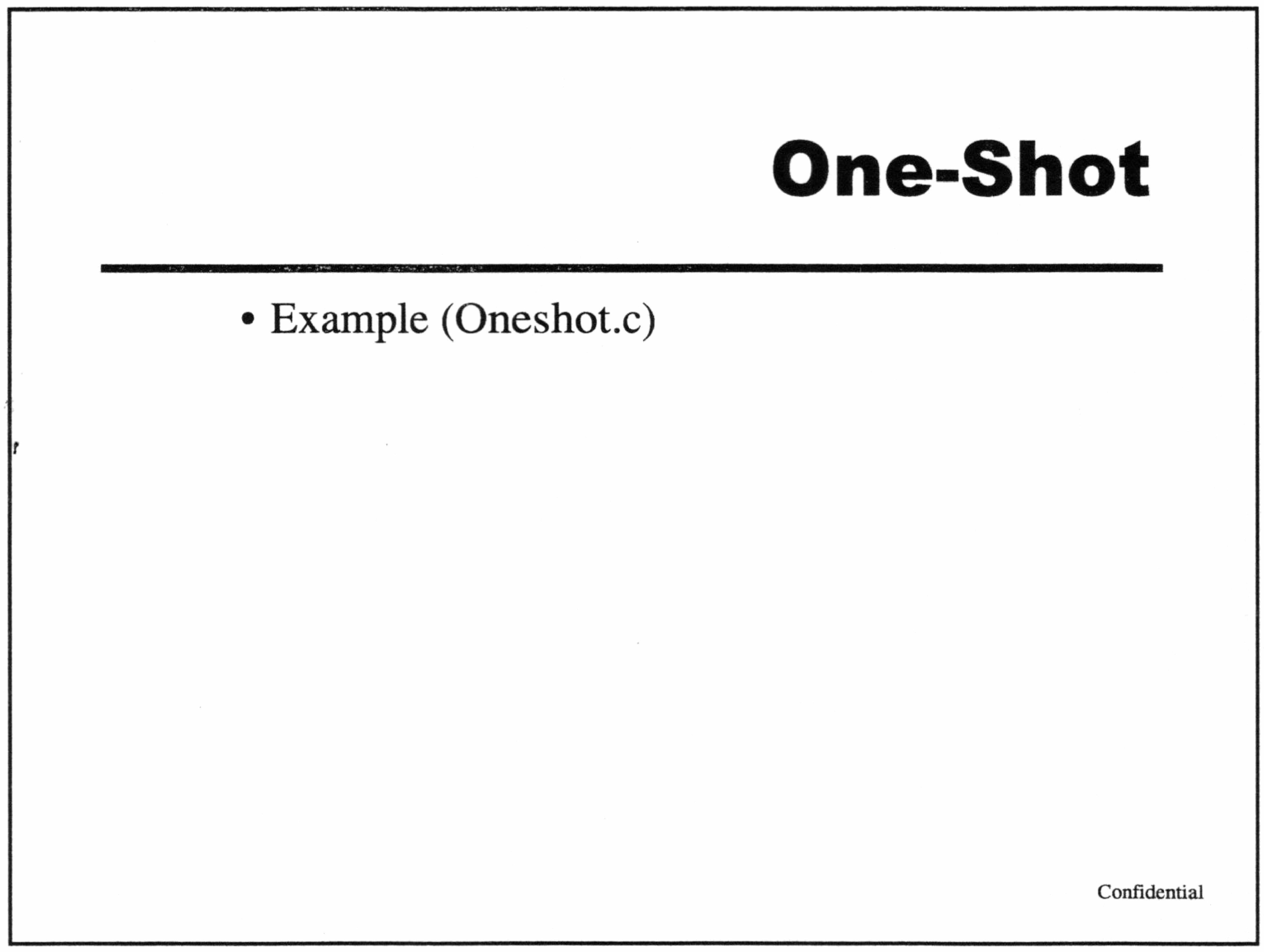
- Playback
  - – Open/close ports
  - – start sounds
  - – set parameters on a per port basis



One-Shot

- Considerations
  - loading a new bank is not instantaneous
  - a 22Khz effect is twice as long as a 44Khz effect, etc. (64K sample limit)
  - small load, mostly on the sound subsystem, not main memory or CPU,
  - No time lag
  - can only play entire sound can't start in middle can't stop in middle
  - 8 sounds can play simultaneously

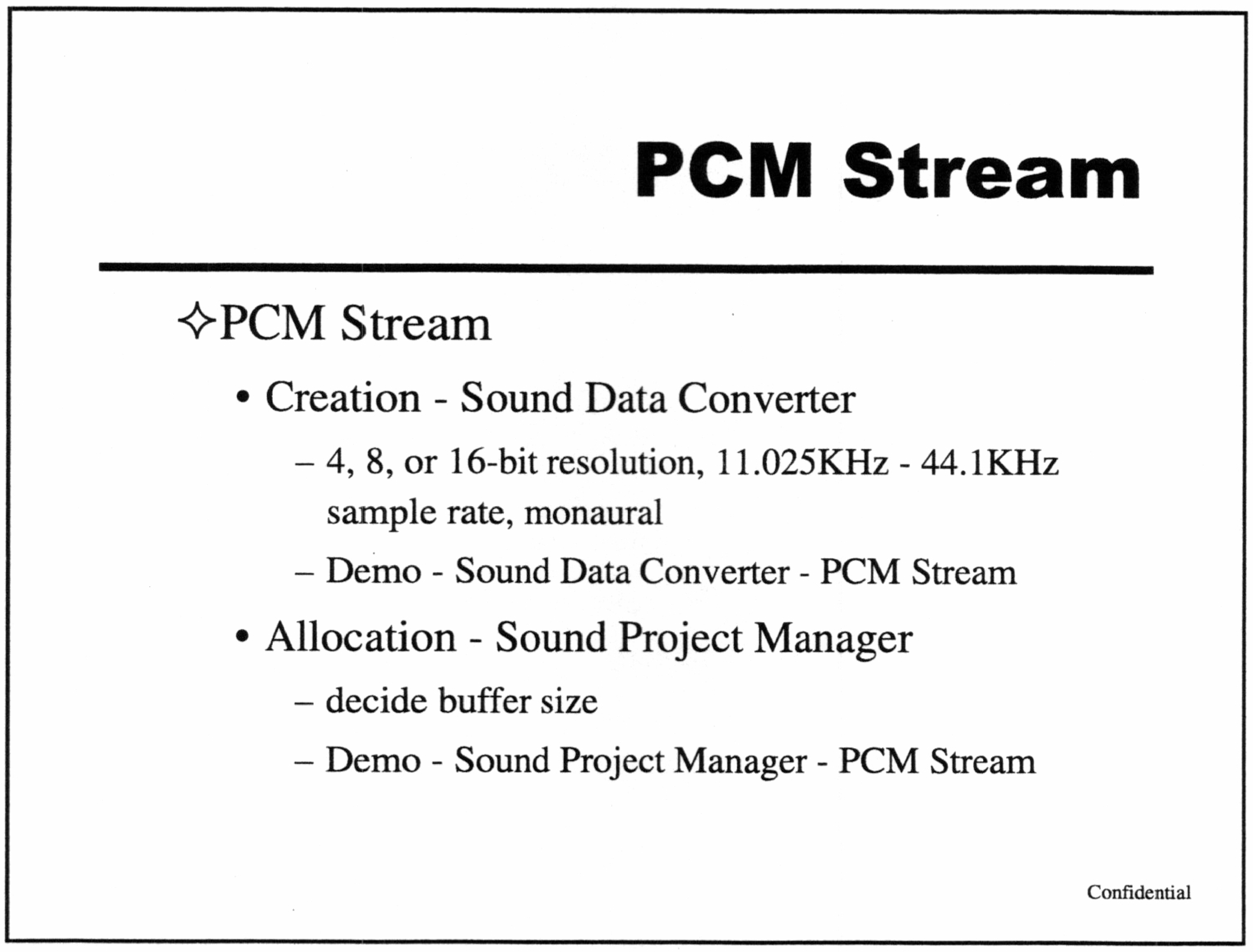
Confidential



One-Shot

- Example (Oneshot.c)

Confidential

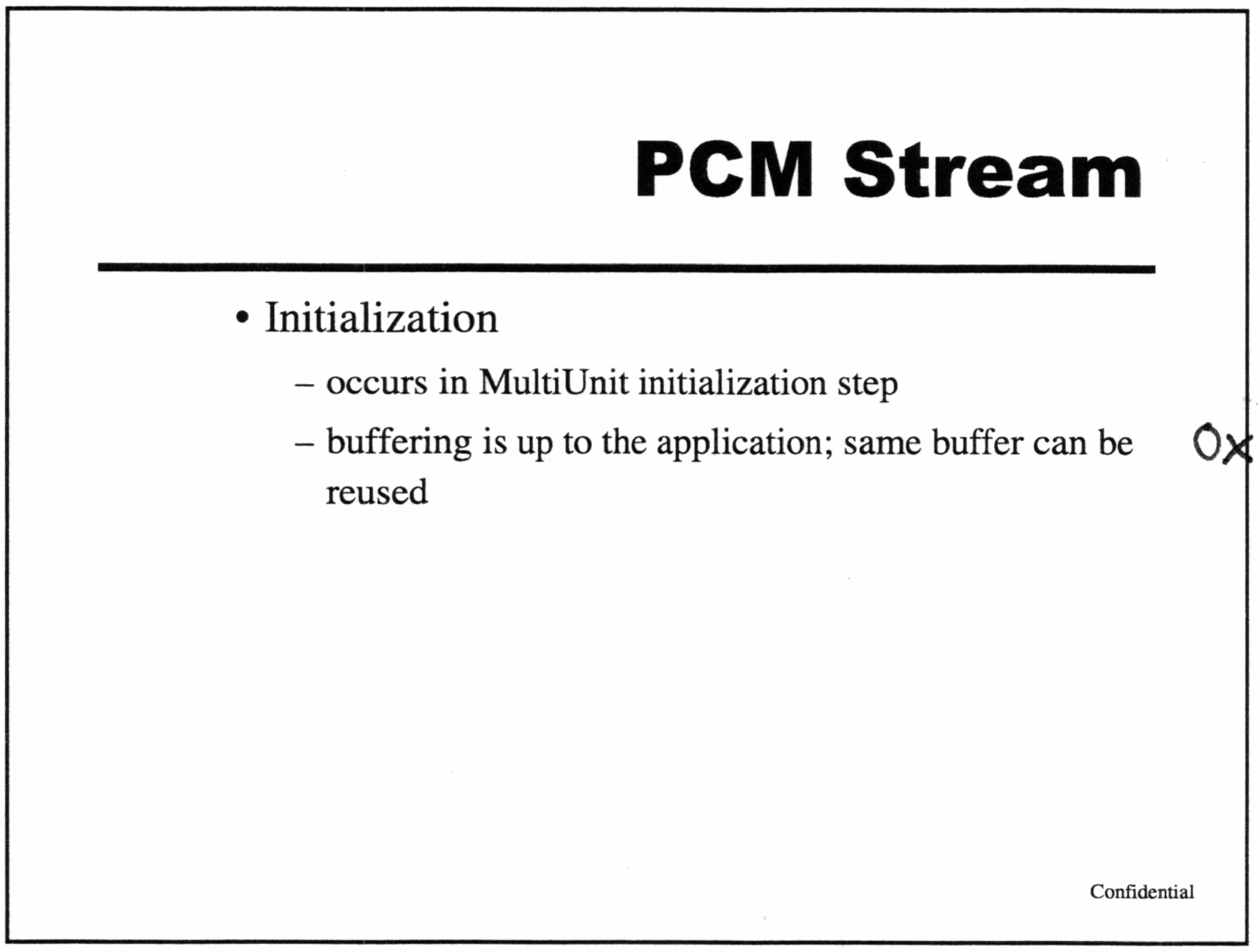


PCM Stream

✧PCM Stream

- Creation - Sound Data Converter
  - – 4, 8, or 16-bit resolution, 11.025KHz - 44.1KHz sample rate, monaural
  - – Demo - Sound Data Converter - PCM Stream
- Allocation - Sound Project Manager
  - – decide buffer size
  - – Demo - Sound Project Manager - PCM Stream

Confidential

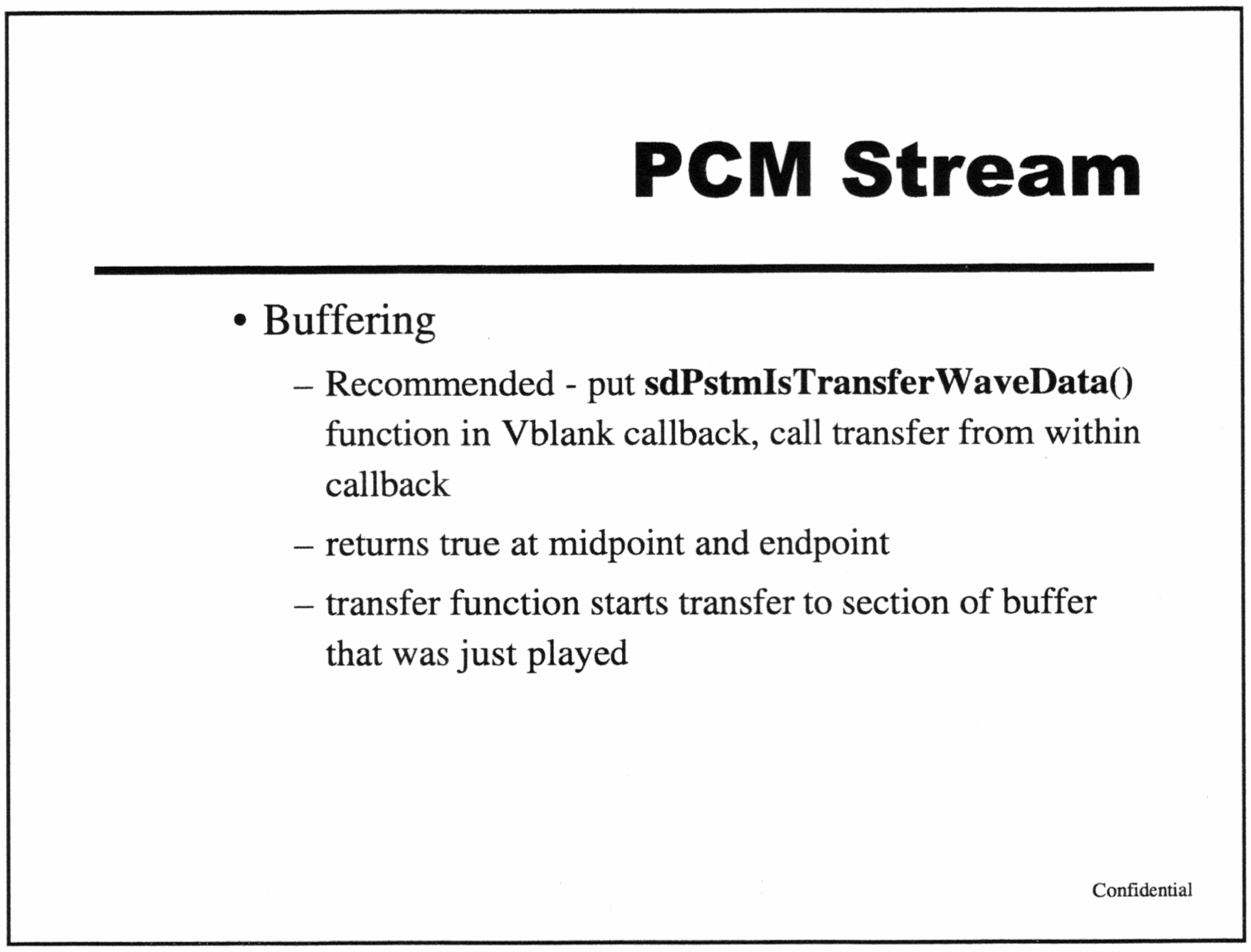


PCM Stream

- Initialization
  - – occurs in MultiUnit initialization step
  - – buffering is up to the application; same buffer can be reused

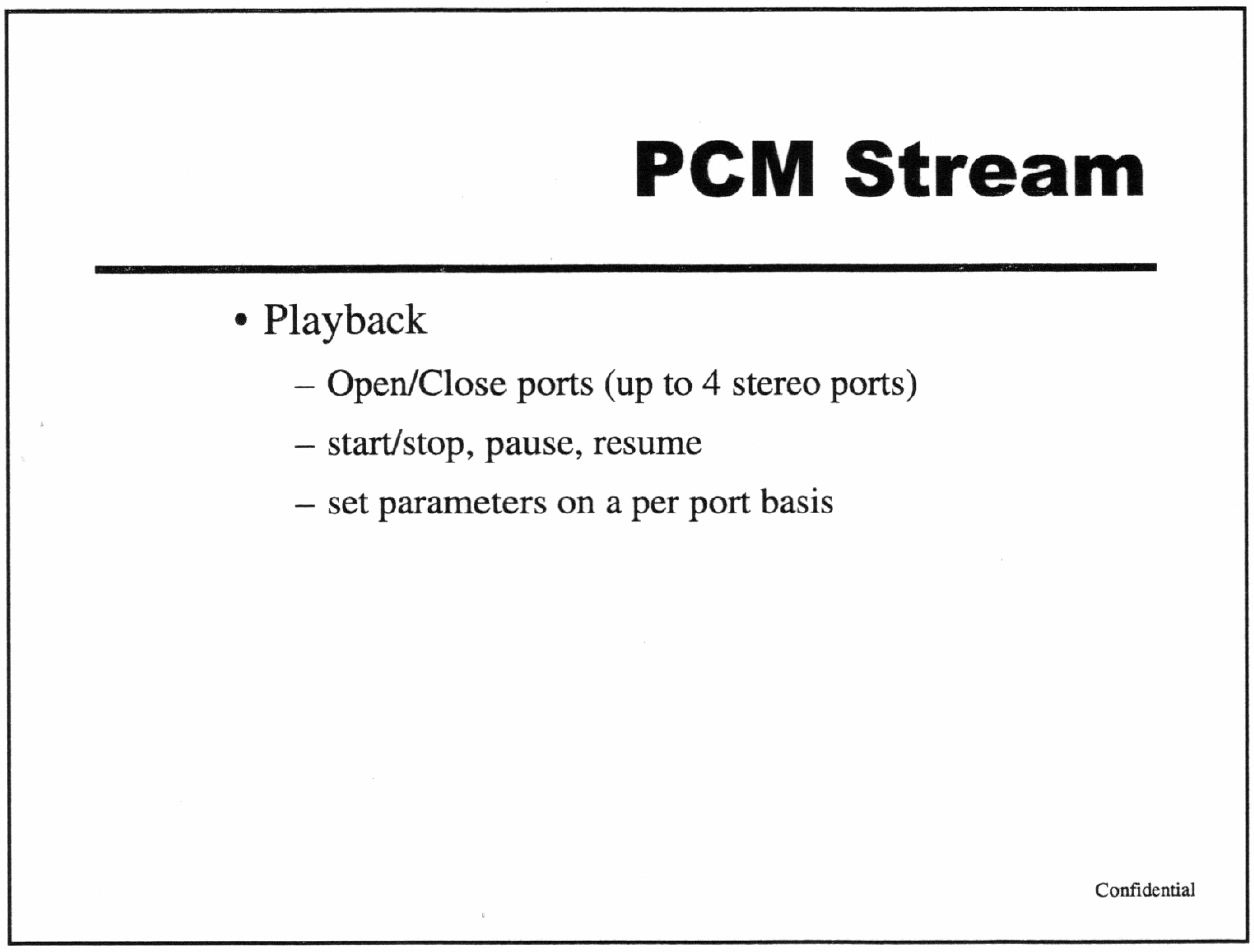
Confidential

0x 1000 is good size



PCM Stream

- Buffering
  - Recommended - put **sdPstmIsTransferWaveData()** function in Vblank callback, call transfer from within callback
  - returns true at midpoint and endpoint
  - transfer function starts transfer to section of buffer that was just played



PCM Stream

- Playback
  - Open/Close ports (up to 4 stereo ports)
  - start/stop, pause, resume
  - set parameters on a per port basis

PCM Stream

- Performance considerations
  - SH4 does most of the work - test to ensure that you don’t drop framerate
  - size of sound memory buffer directly affects SH4 load; recommended value is 1000h or more
  - slight delay before sound plays
  - up to 3 or 4 tracks simultaneous playback maximum before SH4 is slowed down

stero

PCM Stream

- Example (PCMStrm.c)

MIDI

✧MIDI

- Creation (MIDI program bank)
  - specifications and lowest level of bank is very similar to one shot bank
  - loaded with sound effects or a mix of both MIDI and sound effects
  - demo - MIDI Program Editor
  - demo - Sound Project Manager

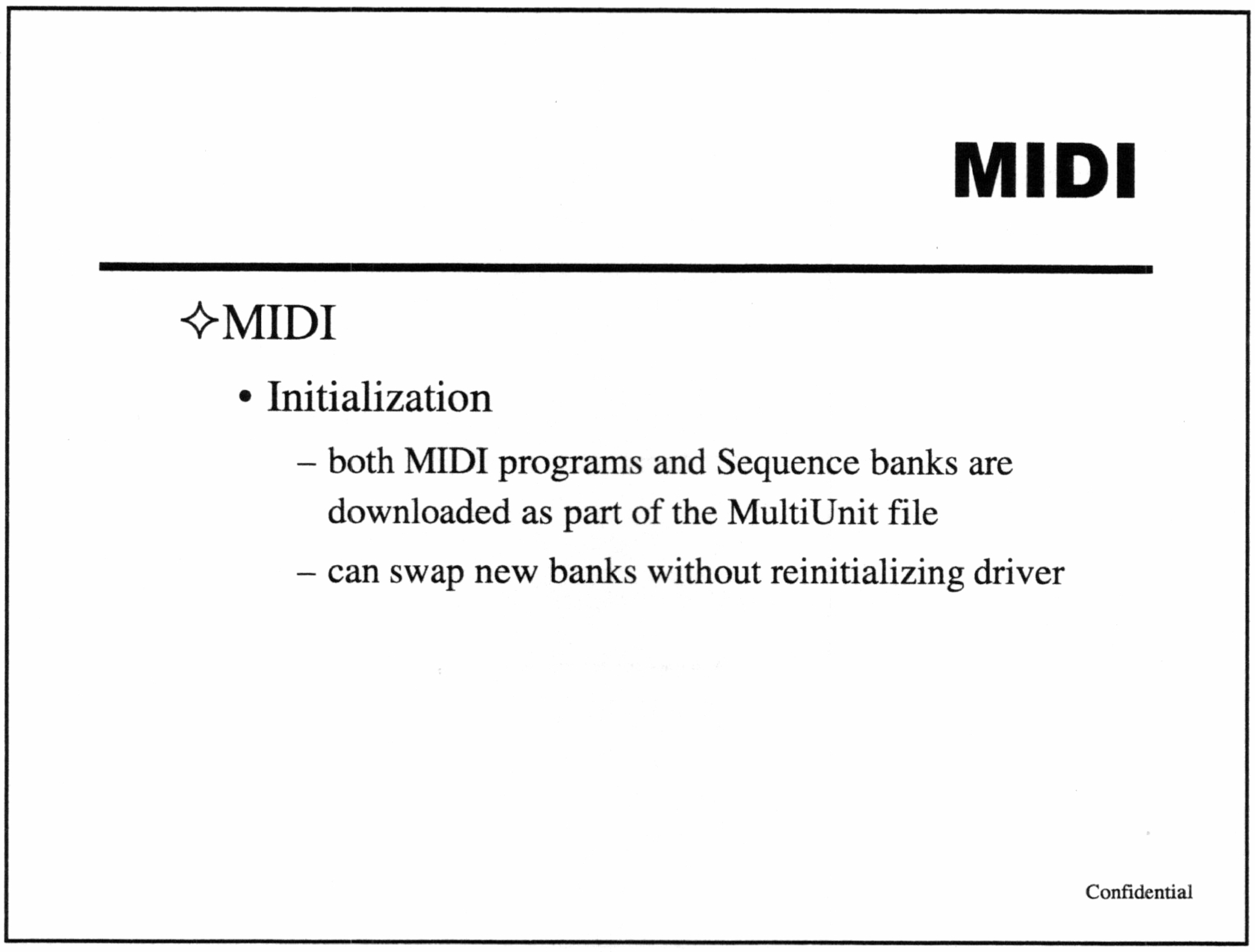
Confidential

MIDI

✧MIDI

- Creation (MIDI Sequence Converter)
  - batch together MIDI sequence (SMF files)
  - demo - Sound Data Converter
  - demo - Sound Project Manager

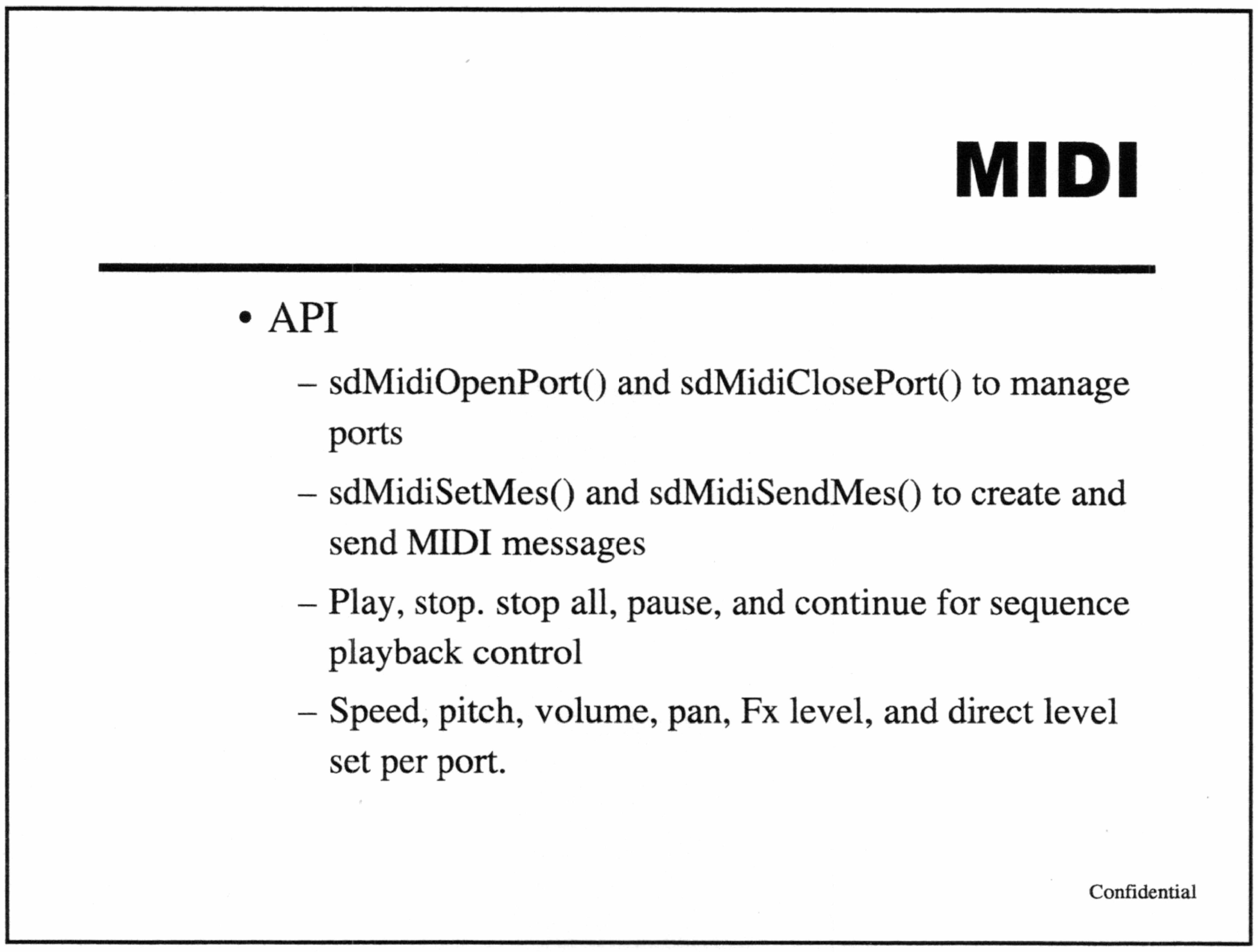
Confidential



MIDI

✧MIDI

- Initialization
  - both MIDI programs and Sequence banks are downloaded as part of the MultiUnit file
  - can swap new banks without reinitializing driver



MIDI

- API
  - sdMidiOpenPort() and sdMidiClosePort() to manage ports
  - sdMidiSetMes() and sdMidiSendMes() to create and send MIDI messages
  - Play, stop. stop all, pause, and continue for sequence playback control
  - Speed, pitch, volume, pan, Fx level, and direct level set per port.

MIDI

- Considerations
  - make sure first MIDI command is always a program change, or a sine wave will play (MIDI sequence or direct messages)
  - low CPU usage, but heavy sound CPU usage to play MIDI sequences - could bog down on dense MIDI sequences - possible stuck notes
  - Most characteristics are the same as one-shot

MIDI

- Examples
  - Direct MIDI playback (MIDIdir.c)
  - Sequence playback (MIDIseq.c)

Sound Effects

Sound effects

- Effect modules
  - Stereo reverb, ERS, stereo delay, pitch shift, stereo chorus, stereo flanger, stereo symphony, and stereo surround
  - Algorithms can be ordered in any way, serial or parallel
  - send and return levels on each in/out channel
  - Qsound

Confidential

Sound Effects

- Using effects
  - Assign One-Shot or PCM stream ports to available FX channels, set wet/dry levels
  - MIDI FX port/channel assignment and levels are determined in the tonebank; generally common to all ports, and levels can be managed by the application
  - GD-DA are routed to FX channels 16 and 17, and are pre-mixed (check this)

Confidential

Sound Effects

- Using effects - Qsound
  - assign ports to Qsound channels
  - application sets position using **sdQsndSetPos()** (32 positions in a flat 180° arc)
  - designed for specific speaker placement, test with different speaker setups

Sound Effects

- Example
  - DSP effects (DSPfx.c)

Dreamcast Audio

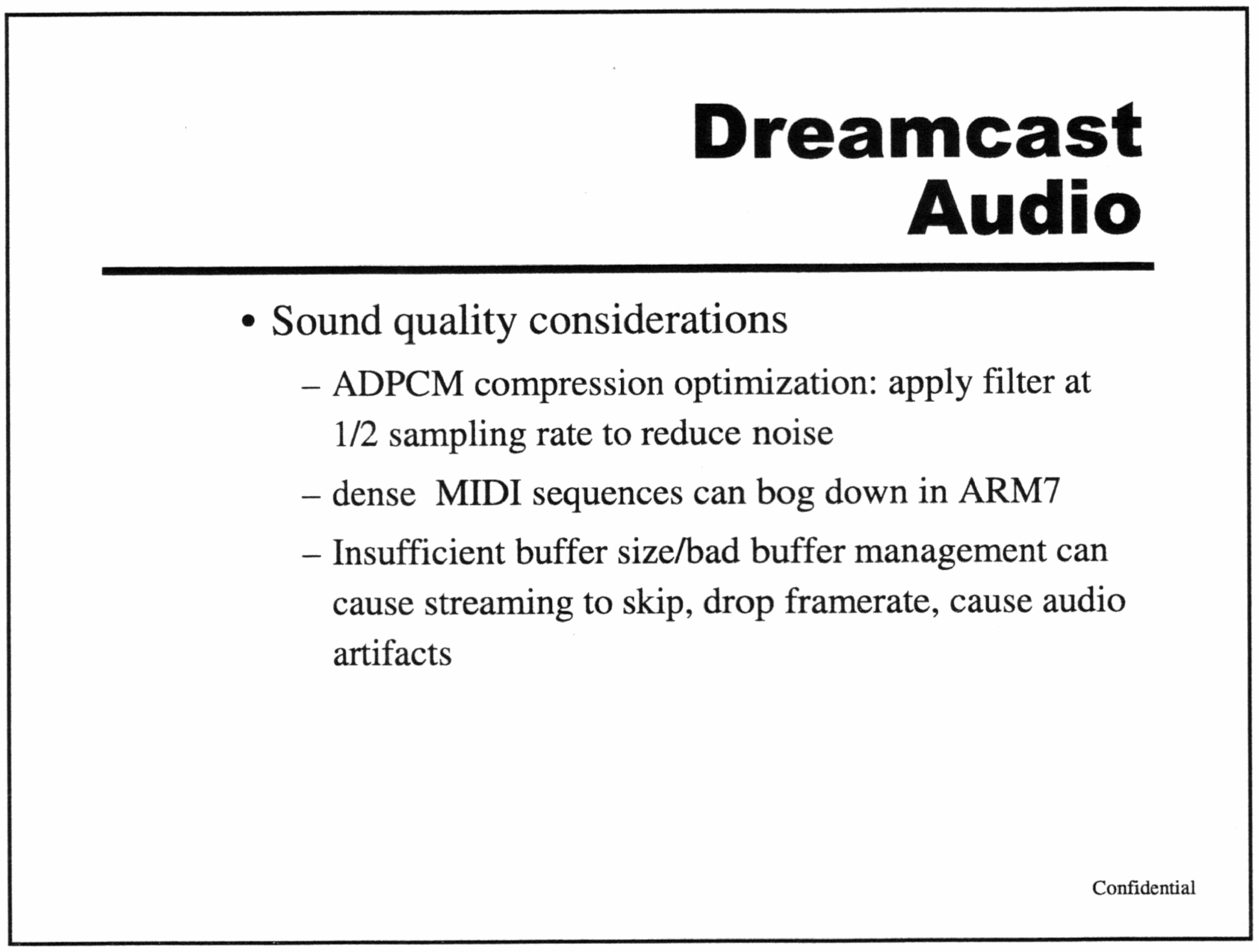
Summary

✧ Hints and tips

- Creating sounds
  - Record as hot as possible to reduce noise, use levels in tools to control actual volume
  - Drop sampling rate when feasible for longer sounds
  - Use envelopes to create complex sounds from simple waveforms

Dreamcast Audio

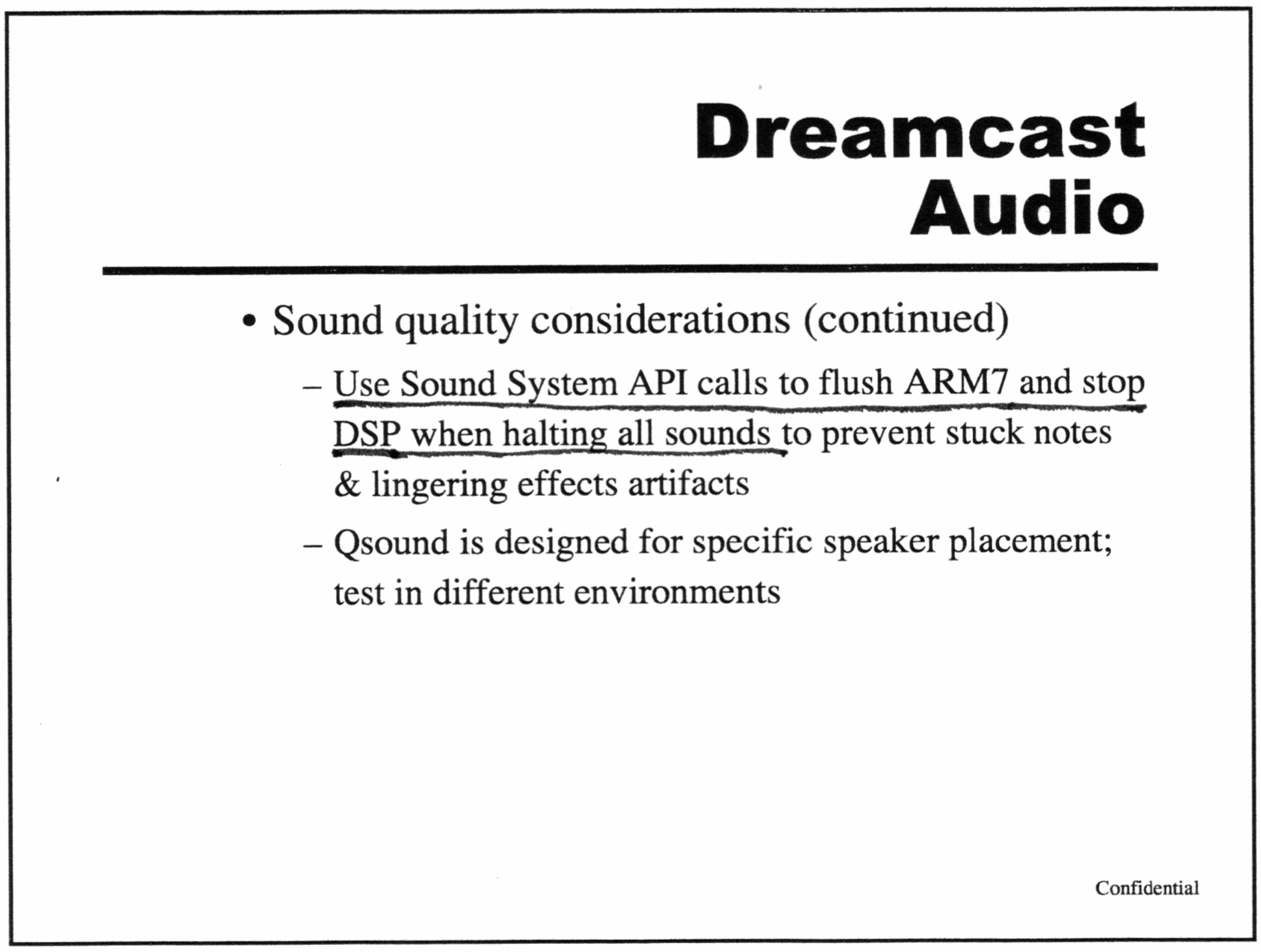
- Implementation notes
  - Maximize use of sound memory; design MultiUnits carefully; keep bank swapping to a minimum (all sounds must temporarily stop to switch banks)
  - play with volume, pitch, etc.; vary parameters to make separate instances of the same sound different for each character or event



Dreamcast Audio

- Sound quality considerations
  - ADPCM compression optimization: apply filter at 1/2 sampling rate to reduce noise
  - dense MIDI sequences can bog down in ARM7
  - Insufficient buffer size/bad buffer management can cause streaming to skip, drop framerate, cause audio artifacts

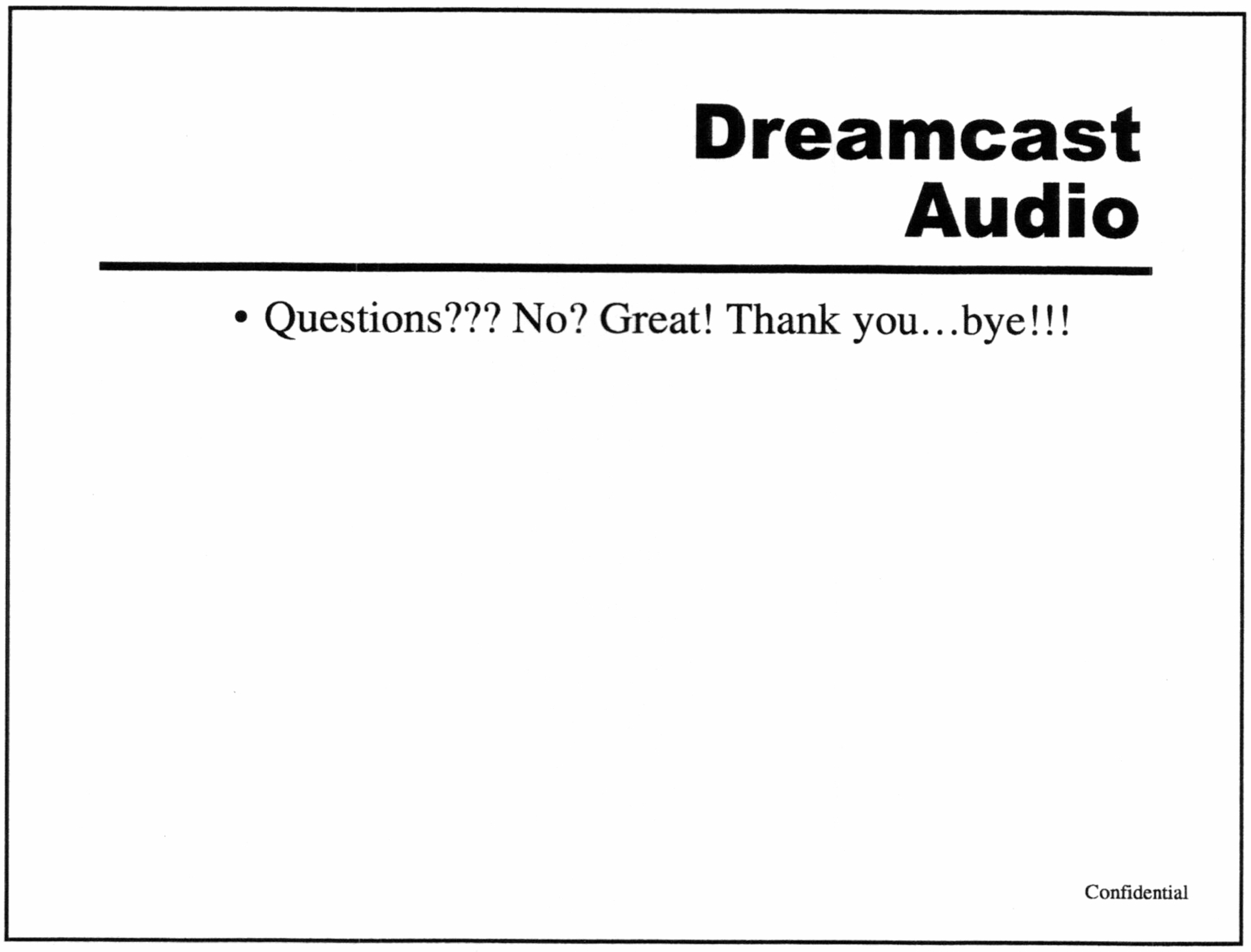
Confidential



Dreamcast Audio

- Sound quality considerations (continued)
  - Use Sound System API calls to flush ARM7 and stop DSP when halting all sounds to prevent stuck notes & lingering effects artifacts
  - Qsound is designed for specific speaker placement; test in different environments

Confidential



Dreamcast Audio

- Questions??? No? Great! Thank you…bye!!!