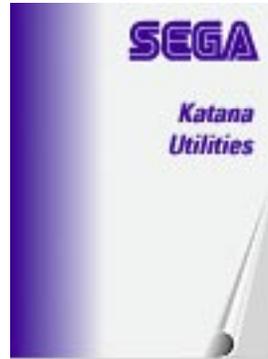




Katana Release 2 Documentation



*Click on an icon to open a book, or
proceed within this document to view TOCs.*

[Getting Started with Katana](#)

Blue links will open the document.

Hardware and Software Requirements

Programming environment:

Supported art packages:

Katana System Components

Installing the Hardware

Installing the Development Software

Manual Installation of the Development Software

Installing the Drivers

First Time Installation

Updating Drivers

Troubleshooting

Video Output

Video Modes

For NTSC (60 Hz):

For PAL (50 Hz):

For VGA:

Building the Samples

Compiling via a DOS window

Compiling within Microsoft Developer Studio

Appendix A - ARC1A Drivers

[Katana Ninja Guide](#)

Blue links will open the document.

01. View Function

Initialization Method

1. Use `njInitView()`.
Set the view as follows:
2. Directly set VIEW structure members.
 1. When performing relative operations:
 2. When performing absolute operations:

View Movement and Rotation

Example:

Structures

02. Texture Guide

Modification History

Ver.0.04

Ver.0.05

ver0.06

Overview

Terminology

Textures

Texture List

Texture Number

Global Index Number

Current Texture List

Current Texture

- PVR Format
- U, V Coordinates
- Aspect Ratio
- Mipmap
- LOD (level of detail)
- Texture Memory
- Cache
- Texture Information Area
- Cache Information Area
- Category Code
- Stride Value

Creating Textures

- Overview

- PVR Format

- Category Code

 - Twiddled, Twiddled Mipmap Format

 - VQ,VQ Mipmap Format (Vector Quantization)

 - Palettize4, Palettize4 Mipmap Format, Palettize8, Palettize8 Mipmap Format

 - Rectangle Format

 - Stride Format

- Color Format

 - Normal Texture Color Format

 - YUV422 format

 - Bump format

 - ARGB8888 format

 - Texture formats supported by NINJA

Memory

- Overview

- Texture Memory

 - Cache

Loading Textures

Overview

Flowchart of Texture Loading

Creating a Texture List

Global Index Number

Texture Load Error

NJD_TEXERR_FILEOPEN

NJD_TEXERR_EXTND

NJD_TEXERR_HEADER

NJD_TEXERR_FILELOAD

NJD_TEXERR_SURFACE

NJD_TEXERR_MAINMEMORY

NJD_TEXERR_TEXMEMLOAD

Global Index Error

Frame Buffer Texture

Texture Functions, Structures, and Definitions

Overview

njInitTexture

njLoadTexture

njLoadTextureNum

njSetTexture

njSetTextureNum

njSetTextureNumG

njLoadCacheTextureNum

njLoadCacheTextureNumG

njReleaseTextureAll

njReleaseTexture
njReleaseTextureNum
njReleaseTextureNumG
njReleaseCacheTextureAll
njReleaseCacheTextureNum
njReleaseCacheTextureNumG
njGetTextureNumG
njCalcTexture
njExitTexture
njSetTexturePath
njSetTextureInfo
njSetTextureName
njReloadTextureNum (New Function)
njReloadTextureNumG (New Function)
njSetRenderWidth (New Function)
NjFrameBufferBmp (New Function)

Deleted functions

Texture Structures

Texture Definitions

Used with nWidth, nHeight

Used with attr

Used with Type, color format

Category code

Texture error code (new addition)

Acquire texture memory size (used with njCalcTexture)

Sample Program

Overview

Sample

Ex. 1 : Display of PVR texture file

Ex. 2: Load a texture from memory and display it.

Ex. 3: Load file from cache and display texture

03. NINJA LIGHT

How to set LIGHT

```
>> void njCreateLight(NJS_LIGHT*, Int)
>> void njDeleteLight(NJS_LIGHT*)
>> void njLightOff(NJS_LIGHT*)
>> void njLightOn(NJS_LIGHT*)
>> void njMultiLightMatrix(NJS_LIGHT*, NJS_MATRIX*)
>> void njSetLight(NJS_LIGHT*)
>> void njSetLightAlpha(NJS_LIGHT*, Float)
>> void njSetLightAngle(NJS_LIGHT*, NJS_Angle, NJS_Angle)
>> void njSetLightColor(NJS_LIGHT*, Float, Float, Float)
>> void njSetLightDirection(NJS_LIGHT*, Float, Float, Float)
>> void njSetLightIntensity(NJS_LIGHT*, Float, Float, Float)
```

```
>> void njSetLightLocation(NJS_LIGHT*, Float, Float, Float)
>> void njSetLightRange(NJS_LIGHT*, Float, Float)
>> void njSetUserLight(NJS_LIGHT*, NJF_LIGHT_FUNC*)
>> void njUnitLightMatrix(NJS_LIGHT*)
>> void njTranslateLightV(NJS_LIGHT*, NJS_VECTOR*)
>> void njTranslateLight(NJS_LIGHT*, Float, Float, Float)
>> void njRotateLightX(NJS_LIGHT*, NJS_Angle) N
>> void njRotateLightXYZ(NJS_LIGHT*, NJS_Angle, NJS_Angle,
>> void njRotateLightY(NJS_LIGHT*, NJS_Angle)
>> void njRotateLightZ(NJS_LIGHT*, NJS_Angle)
```

Macro

How to Use

Example1

Example2

Example3

LIGHTstructure Specification

The members of NJS_LIGHT structure

The members of NJS_LIGHT_ATTR structure

(Example:Spot light)

(Example:point light source)

The members of NJS_LIGHT_CAL structure

04. Ninja Model and Motion

Model Structures

- Diagram of Structure
- Description of Structures
 - Float, Angle
 - Object structure
 - Explanation of evalflags
 - Point structure
 - Texture name structure
 - Texture list structure
 - Material structure
 - Meshset structure
 - Model structures

Construction of a Model

- Diagram of Structure
- Meshsets
 - For a triangular polygon list (NJD_MESHSET_3)
 - For a quadrilateral polygon list (NJD_MESHSET_4)
 - For a contiguous polygon list (NJD_MESHSET_TRIMESH)
 - For an N-sided polygon list (NJD_MESHSET_N)

Construction of a Texture

- Diagram of Structure
 - Overview of Texture Processing
 - Memory-type Textures and the Texture Cache
 - Explanation of the structure of textures
- Ninja Attributes
- Texture Format

Motion Structures

- Diagram of Structure
- Explanation of Structures
 - Action structure
 - Motion structure
 - NJS_MDATA1 to 4 structures
 - Key structures

Construction of a Motion

- Diagram of Structure
- Explanation of Structure

05. Scroll Guide

Revision Information

- Ver.0.04
- Ver.0.05

Image Units as Related to Scrolling

- Overview
- Image Units
 - Pixel
 - Cell
 - Map

Scroll Rotation, Resizing, and Movement

- Overview
- Scroll Rotation, Resizing, and Movement

Scroll Programming

- Overview
- Example of Programming a Scroll

Color

Overview

Color Mode

NJD_COLOR_MODE_FLAT_TEXTURE

NJD_COLOR_MODE_FLAT_TEXTURE_TRANS

Scroll function, Structures, and Definitions

Overview

Scroll-related Functions

njDrawScroll

Scroll-related Structure

Scroll-related Definitions

Texture Structures for Use in Cell Programming

06. Nindows Tutorial

1. Nindows Tutorial

1.1 Summary

1.2 Special Features of Nindows

2. Creating a Simple Nindows Application

2.1 Integrating Nindows

Preparing the Ninja application.

Include the Nindows header file

Call the Nindows initialization function

Call the function to execute Nindows

Call the function to exit Nindows

Linking the Nindows Library

2.2 Description of Functions used in Integrating Nindows

3. Using Nindows and Nindows Utilities

3.1 Using Nindows

System Menu

3.2 Nindows Utilities

Ninja Info Window

Texture Viewer Window

Peripheral Info Window

Window Info Window

Debugging Window

Performance Meter Window

Changing Fonts

4. Windows

4.1 Summary

4.1.1 Types of Windows and Window Classes

4.2 Creating a Window

4.3 Creating a Child Window

4.4 Window Related Parameters

4.5 Description of Window Support Functions

Nindows API

Callback Functions

4.4 Samples and a Description of Window Support Functions

Nindows API

Structure

5. Scroll Windows

5.1 Summary

5.2 Creating a Scroll Window

5.3 Description of Functions Used to Create a Scroll Window

Nindows API

6. Edit Windows

- 6.1 Summary N
- 6.2 Creating and Using an Edit Window
- 6.3 Description of Functions Used in Creating Edit Windows
 - Nindows API
- 6.4 Description of Functions Used in Nindows' Debug Window Utility

7. Scrollbar Controls

- 7.1 Summary N
- 7.2 Creating Scrollbar Controls
- 7.3 Description of Functions Used in Creating Scrollbar Controls
 - Nindows API
 - Structure
- 7.4 Creating Scrollbar Controls that Use Low-level Scrollbar Functions
- 7.5 Description of Low-level Scrollbar Functions
 - Nindows API

8. Button Controls

- 8.1 Summary
- 8.2 Creating a Button Control
- 8.3 Button Validity and Invalidity
- 8.4 Description of Functions for Button Controls
 - Nindows API
 - Callback Function

9. Menus

- 9.1 Summary
- 9.2 Creating and Entering Menu Tables
- 9.3 Menu Callback Functions
- 9.4 Checkmarks
 - Display checkmark
 - Hide checkmark

9.5 Description of Functions for Entering User Menus

Windows API

Callback Function

Structure 2

9.6 Creating Popup Menus

9.6.1 Creating a Simple Popup Menu

9.6.2 Creating a Popup Menu that Stays on the Screen

9.7 Description of Functions Used in Creating Popup Menus

10. Mouse

10.1 Summary

10.2 Getting Mouse Information

10.3 Description of Functions Used for Getting Mouse Information

Windows API

Structure

11. Fonts

11.1 Overview

11.2. Description of Font Functions

Windows API

11.3. Problems with Changing Fonts

Katana Ninja Reference

Blue links will open the document.

System Functions

njAlphaMode Sets Alpha Mode.
njColorBlendingMode Sets Color Blending Mode.
njExitSystem Performs system termination processing.
njIgnoreTextureAlphaMode Sets Texture Alpha Mode.
njInnitSystem Initializes the system.
njInnitVertexBuffer Allocates the buffers for registration of vertex data.
njMipmapAdjust Adjusts Mipmap Level of Textures.
njModifierVolumeMode Sets Modifier Volume Mode
njPolygonCullingMode Sets Polygon Culling Mode
njPolygonCullingSize Sets Polygon Size for Culling
njSetBackColor Sets background color.
njSetVSyncFunction Registers the vertical sync interrupt callback function.
njSpecularMode Sets Specular Mode
njSuperSampleMode Sets texture super sample.
njTextureClampMode Sets the texture clamp.
njTextureFilterMode Sets the texture filter.
njTextureFlipMode Sets the texture flip.
njTextureShadingMode Sets Texture Shading Mode
njVersion Gets the library version.
njWaitVSync Waits for a vertical interrupt.

 **These functions will be deleted in the future**

Matrix Functions

- njAddMatrix**Performs matrix addition.
- njAddVector**Performs vector addition.
- njCalcPoint**Applies matrix conversion to an arbitrary point.
- njCalcVector**Applies matrix conversion to an arbitrary vector.
- ◆ **njClearMatrix**Clears the matrix stack.
- njDetMatrix**Determines a matrix expression.
- njGetMatrix**Gets a copy of the current matrix.
- njInitMatrix**Initializes the matrix stack.
- njInnerProduct**Obtains the inner product of two vectors.
- njInvertMatrix**Obtains the inverse (reversed rows/columns) of a matrix.
- njMirror**Obtains the mirror image of an arbitrary boundary surface.
- njMultiMatrix**Performs matrix multiplication.
- njOuterProduct**Obtains the outer product of two vectors.
- njPopMatrix**Pops the matrix stack.
- njProject**Throws a parallel project onto an arbitrary picture plane.
- njProject2**Projects a transparent view onto an arbitrary picture plane.
- njProjectScreen**Projects an arbitrary point onto the screen.
- njPushMatrix**Pushes the matrix stack.
- njResMatrix**(Unsupported)
- njRotate**Rotates a matrix around an arbitrary axis.
- njRotateX**Applies a matrix that gives a rotation around the X axis.
- njRotateY**Applies a matrix that gives rotation around Y axis.
- njRotateZ**Applies a matrix that gives rotation around Z axis.
- njRotateXYZ**Applies a matrix that gives rotation around X, Y, and Z axes.
- njScale**Scales a matrix.
- njScaleV**Scales a matrix.
- njScalor**Returns the scalar of an arbitrary vector.

njScalor2Returns the square of the scalar of an arbitrary vector.
njSetMatrixCopies an arbitrary matrix.
njSubMatrixPerforms matrix subtraction.
njSubVectorPerforms vector subtraction.
njTranslateApplies a matrix that gives parallel translation along each axis.
njTranslateVMoves a matrix laterally.
njTransposeMatrixTransposes a matrix.
njUnitMatrixConverts an arbitrary matrix to a unit matrix.
njUnitVectorConverts an arbitrary vector to a unit vector.

Collision Functions

njCollisionCheckBBChecks Collision for 2 Hexahedron.
njCollisionCheckBCChecks collision for a hexahedron and a capsule.
njCollisionCheckBSChecking for collision for a hexahedron and a sphere.
njCollisionCheckCCChecks collision for two capsules.
njCollisionCheckSCChecks collision for a sphere and a capsule.
njCollisionCheckSSChecks collision for two spheres.
njDistanceL2LReturns the distance between two lines.
njDistanceL2PLReturns the distance between a line and a plane.
njDistanceP2LReturns the distance between a point and a line.
njDistanceP2PReturns the distance between two points.
njDistanceP2PLReturns the distance between a point and a line.
njDistancePL2PLReturns the distance between two planes.
njGetPnaneNormalFinds the vector that is normal to a plane.
njGetPnaneNormal2Finds the vector that is normal to a plane.
njIsParalellL2LReturns whether two lines are parallel. N
njIsParalellL2PLReturns whether a line and a plane are parallel.
njIsParalellPL2PLReturns whether two planes are parallel.

Mathematical Functions

njAbs	>Returns an absolute value.
njArcCos	>Returns an arc cosine (ArcCos).
njArcCosec	>Returns an arc cosecant.
njArcCot	>Returns an arc cotangent.
njArcSec	>Returns an arc secant.
njArcSin	>Returns an arc sine.
njArcTan	>Returns an arc tangent.
njArcTan2	>Returns an arc tangent (ArcTan2).
njCeil	>Returns the smallest integer not less than n (Ceiling Function).
njCos	>Returns a cosine.
njCosec	>Returns the cosecant.
njCosech	>Returns the hyperbolic cosecant.
njCosh	>Returns the hyperbolic cosine.
njCot	>Returns the cotangent.
njCoth	>Returns the hyperbolic cotangent.
njExp	>Returns exponents.
njFloor	>Returns the largest integer not greater than n (Floor Function).
njFraction	>Returns the decimal fraction.
njHypot	>Returns length of a hypotenuse.
njInvertSqrt	>Returns the inverse square root.
njLog	>Returns the natural logarithm.
njLog10	>Returns the base 10 logarithm.
njLog2	>Returns the base 2 logarithm.
njPow	>Returns the power of a number.
njRandom	Generates a random number.
njRandomSeed	Sets the random number seed.
njRoundOff	Rounds down the decimal fraction.

njRoundUpRounds up the decimal fraction.
njSecReturns a secant.
njSechReturns a hyperbolic secant.
njSinReturns a sine.
njSinhReturns a hyperbolic sine.
njSqrtReturns a square root.
njTanReturns a tangent.
njTanhReturns a hyperbolic tangent.

2D Graphics Functions

njDrawCircle2DDraws circles on a 2D screen.
njDrawLine2DDraws lines on a 2D screen.
njDrawPoint2DDraws points on a 2D screen.
njDrawPolygon2DDraws a polygon on a 2D screen.
njDrawTriangle2DDraws triangles on a 2D screen.

3D Graphics Functions

njDrawLine3DDraws lines in 3D space.
njDrawPoint3DDraws points in 3D space.
njDrawPolygon3DDraws a polygon in 3D space.
njDrawTriangle3DDraws triangles in 3D space.

Light Functions

- njSetLightAlpha** Sets the changes of alpha against material, whose light is set by njCreateLight.
- njCreateLight** Defines a light source type and registers a new light.
- njDeleteLight** Deletes a light created by njCreateLight.
- njLightOff** Deactivates a light created by njCreateLight (turns the light off).
- njLightOn**. Activates a light created by njCreateLight (turns the light on).
- njMultiLightMatrix** Multiplies a matrix with a light matrix.
- njRotateLightX** Rotates a light matrix around the X axis.
- njRotateLightXYZ**. Rotates a light matrix around the X, Y, and Z axes.
- njRotateLightY** Rotates a light matrix around the Y axis.
- njRotateLightZ** Rotates a light matrix around the Z axis.
- njSetLightAngle** Sets the limit angle of a light created with njCreateLight.
- njSetLightColor**. Sets the color of light defined by njCreateLight.
- njSetLightDirection** Sets the direction of light defined by njCreateLight.
- njSetLightIntensity**. Sets the intensity of light defined using njCreateLight.
- njSetLightLocation**. Sets the location of light defined by njCreateLight.
- njSetLightRange**. Sets the limit distance of a light created with njCreateLight.
- njSetUserLight** Assigns a user-defined light function to a light.
- njTranslateLight** Applies a matrix that gives parallel translation along each axis.
- njTranslateLightV**. Moves a light matrix laterally according to a directional vector.
- njUnitLightMatrix**. Unitizes a light matrix.

Scroll Functions

- njDrawScroll** Draws a 2D scroll surface.

Modeling Functions

njControl3D Controls the drawing surface for a 3D object.
njDrawModel Draws a model.
njDrawObject Draws an object.
njFastDrawModel Draws Models.
njFastDrawObject Draws objects.
njInit3D Initializing 3D system.
njSetConstantAttr Sets model attributes.
njSetConstantMaterial Sets model material data.
njSetDepthQueue Sets depth queue.
njSimpleDrawModel Draws models.
njSimpleDrawObject Draws objects.

View Functions

njCalcScreen Projects points in 3D space onto the screen, then finds the screen coordinates to which the points are projected.
njClip2D Specifies the drawing area on the screen.
njClipZ Specifies the limit values of near clipping and far clipping.
njForwardViewAbsolute Moves the view location in the direction of the view. (Absolute Move).
njForwardViewRelative Moves the view location along the viewline. (Relative Move).
njInitView Initializes the view.
njLookAtView Changes the view direction towards point (x, y, z).
njLookAtViewV Changes the view direction towards point (x, y, z).
 **njMultiViewMatrix** Multiplies a view by a matrix.
njReturn2BaseView Returns the current view to the base view.
njRotateViewPosXAbsolute Rotates the view location around the X axis. (Absolute Rotation).

- njRotateViewPosXRelative**Rotates the view around the X axis
(Relative Rotation).
- njRotateViewPosYAbsolute**Rotates the view location around the Y axis.
(Absolute Rotation).
- njRotateViewPosYRelative**Rotates the view around the Y axis
(Relative Rotation).
- njRotateViewPosZAbsolute**Rotates the view location around the Z axis.
(Absolute Rotation).
- njRotateViewPosZRelative**Rotates the view around the Z axis
(Relative Rotation).
- ❗ **njRotateViewX**Rotates the view around the X axis (Absolute Rotation).
- njRotateViewXAbsolute**Rotates the view location around the X axis.
(Absolute Rotation).
- njRotateViewXRelative**Rotates the view around the X axis
(Relative Rotation).
- ❗ **njRotateViewXYZ**Rotates the view around the X, Y, and Z axes
(Absolute Rotation).
- njRotateViewXYZAbsolute**Rotates the line of view around the X, Y,
and Z axes. (Absolute Rotation).
- njRotateViewXYZRelative**Rotates the view around the X, Y,
and Z axes (Relative Rotation).
- ❗ **njRotateViewY**Rotates the view around the Y axis (Absolute Rotation).
- njRotateViewYAbsolute**Rotates the view line around the Y axis.
(Absolute Rotation).
- njRotateViewYRelative**Rotates the view around the Y axis
(Relative Rotation).
- ❗ **njRotateViewZ**Rotates the view around the Z axis (Absolute Rotation).
- njRotateViewZAbsolute**Rotates the view around the Z axis.
(Absolute Rotation).
- njRotateViewZRelative**Rotates the view around the Z axis
(Relative Rotation).
- njSetAspect**Sets the screen aspect ratio.
- njSetBaseView**Sets the current view as the base view.

- njSetPerspective**.....Sets the perspective in horizontal direction.
- njSetScreen**Sets the screen.
- njSetScreenDist**.....Sets the distance from the perspective to the screen.
- njSetView**.....Specifies a user-defined view as the current view.
- ❖ **njTranslateView**Translates the view along the X, Y, and Z axes
(Absolute Translation).
- njTranslateViewAbsolute**.....Moves the view location along the X, Y, and Z axes.
(Absolute Move).
- njTranslateViewRelative**Translates the view along the X, Y, and Z axes
(Relative Translation).
- njTranslateViewV**Translates the view along the X, Y, and Z axes
(Absolute Translation).
- njTranslateViewVAbsolute**.....Moves the view location along the X, Y, and Z axes.
(Absolute Move).
- njTranslateViewVRelative**Translates the view along the X, Y, and Z axes
(Relative Translation).
- njUnitBaseViewVector**.....Converts the original view vector to a unit vector.
- njUnitCurrentViewVector**.....Converts the view vector of the current view to a unit vector.
- ❖ **njUnitViewMatrix**Sets a unit matrix to the view matrix.
- njUnitViewVector**Converts the view vector to a unit vector.

Texture Functions

njCalcTextureCalculates the remaining texture memory size.

njGetTextureNumGObtains the global index number of the current texture.

njInitTextureSets the area used for storing texture information.

njLoadCacheTextureSets the cache information area.

njLoadCacheTextureNumLoads a texture by texture number.

njLoadCacheTextureNumGLoads texture number globalIndex from cache memory into texture memory.

njLoadTexture.Loads a texture.

njLoadTextureNumLoads textures.

njReleaseCacheTextureNumRelease cache memory.

njReleaseCacheTextureAllRelease all cache memory.

njReleaseCacheTextureNumGReleases cache memory.

njReleaseTextureReleases texture memory.

njReleaseTextureNumReleases texture memory.

njReleaseTextureNumG.Releases texture memory. N

njReleaseTextureAllRelease all texture memory.

njSetTextureSet current texture list.

njSetTextureInfo.Set data into texture name structure.

njSetTextureName.Set data into texture name structure.

njSetTextureNum.Sets a texture number as the current texture.

njSetTextureNumGSets the current texture to a global index number.

njReloadTextureNumReloads Textures.

njReloadTextureNumGReloads Textures.

njRenderTextureNum.Renders Texture Area. N

njRenderTextureNumGRenders at the Texture Area.

njSetRenderWidthSets Stride Value.

Sprite Functions

njDrawSprite2DDraws 2D sprite.
njDrawSprite3DDraws 3D sprite.

Debugging Functions

njPrintBDisplays a value in binary notation.
njPrintCDisplays a character string.
njPrintColor.....Specifies the character color.
njPrintDDisplays a value in decimal notation.
njPrintFDisplays a floating point decimal value.
njPrintHDisplays a value in decimal notation. N
njFrameBufferBmp.....Converts Frame Buffer To Bit Map Image.
njPrintSize.....Specifies Character Size.

Special Effects Functions

njExcuteFadeExcutes the fade effect.
njFadeDisableDisables the fade effect.
njFadeEnableEnables the fade effect.
njFogDisableDisables the fog effect.
njFogEnableEnables the fog effect.
njGenerateFogTable.....Creates Fog Table.
njGenerateFogTable2.....Creates Fog Table and Sets Density.
njGenerateFogTable3.....Creates Fog Table and Sets Density.
njSetFadeColor.....Specifies the fade color.
njSetFogColor.....Specifies the fog color.
njSetFogDensity.....Specifies the fog density.
njSetFogTable.....Sets a user-defined fog table.

Motion Functions

njActionDrawing motion.
njActionLinkLinks motions.
njDrawMotionDrawing Motion.
njDrawMotionLinkLinks motions.
njDrawShapeMotionExecutes motion that includes shapes.
njDrawShapeMotionLinkLinks motions that include shapes.
njFastActionDraws Motion.
njFastActionLinkLinks motions.
njFastDrawMotionDraws Motion.
njFastDrawMotionLinkLinks motions.
njFastDrawShapeMotionExecutes motion that includes shapes.
njFastDrawShapeMotionLinkLinks motions that include shapes.
njSetMotionCallbackRegistering motion callback routine.

Memory Functions

njMemCopyMemory Copy.
njMemCopy4Memory Copy. (Word)
njMemCopy2Memory Copy. (Long)

Drawing Functions

njDrawPolygonDraws polygons without textures.
njDrawTextureDraw texture polygons.

Input Functions

njGetPeripheralGets information of input device. (Peripheral).
njPrintPeripheralInfoDisplays peripheral condition.

Kamui

Blue links will open the document.

1. OVERVIEW

2. BASIC PROCESSING FLOW

3. KAMUI FUNCTIONS

3.1 NAMING RULES

3.2 INITIALIZATION FUNCTION

3.2.1 Initializing the Rendering Chip

3.3 SURFACE HANDLING FUNCTION

3.3.1 Setting the Display Mode

3.3.2 Creating the Primary Surface and Off-Screen Surface

3.3.3 Creating the Texture Surface

3.3.4 Creating the Texture Surface for Mipmap/VQ Texture

3.3.5 Creating the Texture Surface in Contiguous Address Areas

3.3.6 Using Frame Buffer as Texture Surface

3.3.7 Setting the a Threshold Value

3.3.8 Determining the Display Screen

3.3.9 Flipping the Display Screen

3.4 SETTING PARAMETERS FOR EACH SCENE SEPARATELY

3.4.1 Setting the Culling Parameter

3.4.2 Setting the Color Clamp Value

3.4.3 Setting the Fog Color

3.4.4 Specifying a Fog Density

3.4.5 Setting the Fog Table

3.4.6 Setting the On-Chip Palette Mode

3.4.7 Setting the On-Chip Palette

3.4.8 Setting the Border Color

3.4.9 Registering the Rendering Parameter of the Background Plane

- 3.4.10 Setting the Background Plane
- 3.4.11 Setting Autosort Mode
- 3.4.12 Specifying Pixel-Unit Clipping
- 3.4.13 Specifying the Stride Size
- 3.4.14 Setting the CheapShadowMode
- 3.4.15 Specifying the Number of VsyncWait States
- 3.4.16 Forced Reset of Renderer

3.5 SETTING PARAMETERS OF EACH VERTEX

- 3.5.1 VERTEXCONTEXT
- 3.5.2 Setting Rendering Parameters for Each Vertex
- 3.5.3 Registering Rendering Parameters for Each Vertex
- 3.5.4 Registering Rendering Parameters of a Modifier Volume
- 3.5.5 Setting Global Clipping
- 3.5.6 Setting a Strip Length
- 3.5.7 Setting a User Clipping Area
- 3.5.8 Setting a User Clipping Area (Direct Specification)
- 3.5.9 Direct Rewrite Mode for Rendering Status

3.6 RECORDING VERTEX DATA

- 3.6.1 Recording Vertex Data
- 3.6.2 Vertex Data Structure
- 3.6.3 Vertex Parameter
- 3.6.4 Setting a Modifier Volume
- 3.6.5 Allocating a Vertex Data Buffer
- 3.6.6 Releasing Vertex Data Registration Buffers
- 3.6.7 Allocating the Tile Accelerator Output Buffer
- 3.6.8 Writing Global Parameters in a Buffer
- 3.6.9 Directly Writing Global Parameters
- 3.6.10 Writing Vertex Data in a Buffer
- 3.6.11 Directly Writing Vertex Data
- 3.6.12 Reporting the End of Vertex Registration

- 3.6.13 Notifying the End of Vertex Data Writing
(When the Data Is Written into Buffers)
- 3.6.14 Notifying the End of Vertex Data Writing
(When Data Is Directly Written)
- 3.6.15 Rendering into the Texture Memory
(When Data Is Written into Buffers)
- 3.6.16 Rendering into the Texture Memory (When Data Is Directly Written)
- 3.6.17 Specifying a Modifier Volume List
- 3.6.18 Obtaining Current Writing Position of VertexBuffer
- 3.6.19 Changing Current Writing Position of VertexBuffer
- 3.6.20 Direct Rewriting of Vertex Control Word
- 3.6.21 Flushing Opaque VertexBuffer

3.7 CALLBACK FUNCTIONS AND CALLBACK AUXILIARY FUNCTIONS

- 3.7.1 Specifying a Rendering End Callback Function
- 3.7.2 Specifying a V-Sync Callback Function
- 3.7.3 Specifying an H-Sync Interrupt Callback Function
- 3.7.4 Setting the H-Sync Interrupt Line
- 3.7.5 Reading the Current H-Sync Line
- 3.7.6 Specifying a Texture Memory Overflow Callback Function
- 3.7.7 Specifying a Strip Buffer Overrun Callback Function
- 3.7.8 Specifying a Vertex Data Transfer End Callback Function

3.8 OTHER FUNCTIONS

- 3.8.1 Stopping the Frame Buffer Display
- 3.8.2 Obtaining the Version Information

3.9 TEXTURE HANDLING FUNCTIONS OF KAMUI

- 3.9.1 Loading Texture Data
- 3.9.2 Re-reading the Code Book Portion of VQ Texture
- 3.9.3 Reloading a Particular Mipmap Texture
- 3.9.4 Reading the YUV-Format Texture Data
- 3.9.5 Deleting Texture Data
- 3.9.6 Obtaining the Available Texture Memory Space

- 3.9.7 Reading the Texture in Texture Memory
- 3.9.8 Garbage Collection of Texture Memory

4. KAMUI UTILITY LIBRARY

4.1 TEXTURE-RELATED FUNCTIONS

- 4.1.1 Conversion from KAMUI Bit Map Format to Twiddled Format
- 4.1.2 Conversion from Frame Buffer Format (Rectangle) to Windows BMP Format

5. STRUCTURES

- 5.1 FRAME BUFFER/TEXTURE SURFACE STRUCTURE
- 5.2 VERSION INFORMATION STRUCTURE
- 5.3 VERTEX CONTEXT
- 5.4 PACKED 32-BIT COLORS
- 5.5 PALETTE DEFINITION STRUCTURE
- 5.6 VERTEX DATA BUFFER STRUCTURE

6. TEXTURE FORMAT

- 6.1 TEXTURE FORMATS SUPPORTED BY ARC1/CLX1
- 6.2 ARC1/CLX1 TEXTURE FORMAT
 - 6.2.1 Texture Format of KAMUI
 - 6.2.2 Twiddled Format and Twiddled Mipmap Format
 - 6.2.3 VQ Format and VQ Mipmap Format
 - 6.2.4 Palettized 4-bpp/8-bpp Format
 - 6.2.5 Rectangle Format
 - 6.2.6 Stride Format
 - 6.2.7 BUMP-Mapping Format
 - 6.2.8 Kamui Bit Map Format

Katana Utilities

Blue links will open the document.

01. PowerVR Texture Converter

How to Use PVRConv

Command Line Usage

Alpha Channel

.bmp:

.pix:

PVRConv Defaults

PVRConv Options

-p <dir> | -path <dir> :
-ns | -nosuffix :
-gi <num> | -globalindex <num> :
-t | -twiddled :
-r | -rectangle :
-s | -stride :
-nm | -nomipmap :
-nd | -nodither :
-na | -noadither :
-5 | -565 :
-4 | -4444 :
-1 | -1555 :
-t1 | -t1555 :
-b4 | -b4444 :
-ra | -ralpha :

PowerVR Texture Format

02. VQ Compression Tool UPC-7

How to Use VQ Compress

Command Line Usage

VQ Compress Defaults

VQ Compress Options

VQ Compression Ratios

03. Using the LightWave Converter

Overview of the LightWave Converter

Output Files

Execution

Path Input

Creating Models for the Converter

Textures

< twiddled format >

<rectangle format>

Texture Extraction (Alpha)

Material Names

<flags>

<Filter mode>

Motion Output Using an MRS File

Creating Shape Data

Restrictions

04. Gigen Tool

Function Overview

Parameters

gigen <target directory name> [option] Execution Method

Options

-b baseNo

-l

-i

05. Ninja Export for 3D Studio MAX R2

Overview

Files used in libraries

Other files

Installation Method

Usage

Options Dialog Box

NSC File (Scene)

Create NSC File

NJA File (Model)

Strip

Add Polygon Normal List

MRS File (Motion Resource)

Create MRS File

PVR File (Texture)

Overwrite Texture File

Texture Global Index Base

Texture Type

Pixel Format

- Mipmap OFF
- Dither OFF
- Alpha Dither OFF
- NAM File (TRS Motion)
 - MRS File
 - Motion Type U
 - SelectKeys
 - OptimizeDataSize
- NAS File (Shape Motion File)
 - Data Type
 - Select Keys
- Global Options
 - XAxis -90 Rotate
 - Create Log File
 - Reset
 - Cancel
 - OK

Models

- Outputting Selected Models
- Materials
- Textures
 - <twiddled format>
 - <rectangle format>
- Texture Extraction (Alpha)
 - Method 1: Extraction by 32-bit TGA
 - Method 2: Extraction by 24-bit BMP + 24-bit BMP
- Texture UV
- TRS Motion
 - Ignore Keys (All Frames)
 - Key Frames

Motion Output Using an MRS File

Shape Motion

 UserKeys (Plugin)

 Modifier Keys

 Ignore Keys (All Frames)

Attributes

 Double-sided Polygons

 Flat Shading

Attributes That Are Set by Adding Characters at the Beginning of Material Names

 <flags>

 <Filter mode>

Error Messages

Other Notes

Future Development

06. KeyListGen (3D Studio Max R2)

Overview

Usage

Restrictions

Future Development

07. NinjaExport for SoftImage3D NT

ninjaExport Overview

ninjaExport Package Contents

ninjaExport Installation method

ninjaExport Functions

Setting for File Output

Options for Models

Texture-related Options

Motion-related Options

ninjaExport Texture

 twiddled format

 rectangle format

ninjaExport Material

 flags

 Filter mode

ninjaExport Future Support Plan

08. NinjaExport for SoftImage SGI

Overview

Flow for the gigen (globalIndex Generator) Command

Notes on the Ninja File Format and Compiling