

PlayStation®Edge DXT Library Reference

Table of Contents

Preface	3
About This Document.....	4
SPU Functions	5
edgeDxtCompress1	6
edgeDxtCompress1a	7
edgeDxtCompress3	8
edgeDxtCompress5	9
edgeDxtDecompress1	10
edgeDxtDecompress3.....	11
edgeDxtDecompress5.....	12

Preface

About This Document

Purpose

This document provides an API reference for the Edge DXT component of the Edge library. Use this component to efficiently perform compression of texture data to the RSX™ compressed formats on SPU.

Audience and Prerequisites

This document was written for PlayStation®3 developers who want to write high-performance applications for the PlayStation®3. It is assumed that such developers have familiarity with the following:

- C and C++
- PlayStation®3 hardware
- SCE standard library functions

Related Documentation

In combination with this reference, the following documents provide complete usage and reference information about the Edge library:

- *PlayStation®Edge Library Overview*
- *PlayStation®Edge Geometry Library: Quick Start*
- *PlayStation®Edge Geometry Library Reference*
- *PlayStation®Edge Geometry Library for Offline Tool: Reference*
- *PlayStation®Edge Animation Library Reference*
- *PlayStation®Edge Animation Library for Offline Tool: Reference*
- *PlayStation®Edge Zlib Library Reference*
- *PlayStation®Edge LZMA Library Reference*
- *PlayStation®Edge LZO Library Reference*
- *PlayStation®Edge Post Library Reference*

Typographic Conventions

This document uses the following typographic conventions:

Convention	Meaning
<code>fixed-width font</code>	Indicates programming code and literals, such as processing instructions, register names, data types, events, and file names. Also indicates function, structure, and macro names.
<code>fixed-width font + bold</code>	In structure/function definitions only, indicates names of structures and functions.
<i>fixed-width font + italics</i>	Indicates arguments, parameters, and variables.
blue + underlined text	Indicates a hyperlink (blue displays in color printers or online only).

SPU Functions

edgeDxtCompress1

Compresses pixel data to the CELL_GCM_TEXTURE_COMPRESSED_DXT1 format.

Definition

```
#include <edge/dxt/edgedxt.h>
void edgeDxtCompress1
(
    void* outputBlocks,
    const void* inputPixels,
    uint32_t rowStride,
    uint32_t blockCount
);
```

Arguments

<i>outputBlocks</i>	Pointer to where the compressed output data will be written to in Local Store. Must be 8-byte aligned.
<i>inputPixels</i>	Pointer to where the input pixel data will be read from Local Store. Must be 16-byte aligned.
<i>rowStride</i>	The number of bytes between each row of input pixel data. Must be 16-byte aligned.
<i>blockCount</i>	The number of 4x4 blocks of pixel data to compress.

Description

This function compresses pixel data to the CELL_GCM_TEXTURE_COMPRESSED_DXT1 texture format within Local Store.

The color components at *inputPixels* are expected to be in ARGB order. For example, non-tiled surface data of type CELL_GCM_SURFACE_A8R8G8B8 can be compressed directly once it has been moved into Local Store. Other formats should be shuffled into ARGB order before compression.

In order to maintain good compression speed, this function always uses the 4-color DXT1 encoding. As such, any alpha data in the input pixels is ignored – the output blocks will always have a constant alpha value of 255. Use the [edgeDxtCompress1a\(\)](#), [edgeDxtCompress3\(\)](#), or [edgeDxtCompress5\(\)](#) compression functions for pixel data where alpha needs to be preserved.

This function will write exactly $8 * \text{blockCount}$ bytes of texture data to the LS address *outputBlocks*.

Notes

If any of the parameter alignment restrictions are violated, the low bits of the parameter will be ignored.

edgeDxtCompress1a

Compresses pixel data to the CELL_GCM_TEXTURE_COMPRESSED_DXT1 format.

Definition

```
#include <edge/dxt/edgedxt.h>
void edgeDxtCompress1a
(
    void* outputBlocks,
    const void* inputPixels,
    uint32_t rowStride,
    uint32_t blockCount
);
```

Arguments

<i>outputBlocks</i>	Pointer to where the compressed output data will be written to in Local Store. Must be 8-byte aligned.
<i>inputPixels</i>	Pointer to where the input pixel data will be read from Local Store. Must be 16-byte aligned.
<i>rowStride</i>	The number of bytes between each row of input pixel data. Must be 16-byte aligned.
<i>blockCount</i>	The number of 4x4 blocks of pixel data to compress.

Description

This function compresses pixel data to the CELL_GCM_TEXTURE_COMPRESSED_DXT1 texture format within Local Store.

The color components at *inputPixels* are expected to be in ARGB order. For example, non-tiled surface data of type CELL_GCM_SURFACE_A8R8G8B8 can be compressed directly after it has been moved into Local Store. Other formats should be shuffled into ARGB order before compression.

This function will generate an output image with 1-bit alpha. The output alpha value will be set to 255 if the input alpha value is greater than or equal to 0x80; otherwise it will be set to 0.

This function will write exactly $8 \times \text{blockCount}$ bytes of texture data to the LS address *outputBlocks*.

Notes

If any of the parameter alignment restrictions are violated, the low bits of the parameter will be ignored.

edgeDxtCompress3

Compresses pixel data to the CELL_GCM_TEXTURE_COMPRESSED_DXT23 format.

Definition

```
#include <edge/dxt/edgedxt.h>
void edgeDxtCompress3
(
    void* outputBlocks,
    const void* inputPixels,
    uint32_t rowStride,
    uint32_t blockCount
);
```

Arguments

<i>outputBlocks</i>	Pointer to where the compressed output data will be written to in Local Store. Must be 16-byte aligned.
<i>inputPixels</i>	Pointer to where the input pixel data will be read from Local Store. Must be 16-byte aligned.
<i>rowStride</i>	The number of bytes between each row of input pixel data. Must be 16-byte aligned.
<i>blockCount</i>	The number of 4x4 blocks of pixel data to compress.

Description

This function compresses pixel data to the CELL_GCM_TEXTURE_COMPRESSED_DXT23 texture format within Local Store.

The color components at *inputPixels* are expected to be in ARGB order. For example, non-tiled surface data of type CELL_GCM_SURFACE_A8R8G8B8 can be compressed directly once it has been moved into Local Store. Other formats should be shuffled into ARGB order before compression.

This function will write exactly $16 * \text{blockCount}$ bytes of texture data to the LS address *outputBlocks*.

Notes

If any of the parameter alignment restrictions are violated, the low bits of the parameter will be ignored.

edgeDxtCompress5

Compresses pixel data to the CELL_GCM_TEXTURE_COMPRESSED_DXT45 format.

Definition

```
#include <edge/dxt/edgedxt.h>
void edgeDxtCompress5
(
    void* outputBlocks,
    const void* inputPixels,
    uint32_t rowStride,
    uint32_t blockCount
);
```

Arguments

<i>outputBlocks</i>	Pointer to where the compressed output data will be written to in Local Store. Must be 16-byte aligned.
<i>inputPixels</i>	Pointer to where the input pixel data will be read from Local Store. Must be 16-byte aligned.
<i>rowStride</i>	The number of bytes between each row of input pixel data. Must be 16-byte aligned.
<i>blockCount</i>	The number of 4x4 blocks of pixel data to compress.

Description

This function compresses pixel data to the CELL_GCM_TEXTURE_COMPRESSED_DXT45 texture format within Local Store.

The color components at *inputPixels* are expected to be in ARGB order. For example, non-tiled surface data of type CELL_GCM_SURFACE_A8R8G8B8 can be compressed directly once it has been moved into Local Store. Other formats should be shuffled into ARGB order before compression.

This function will write exactly $16 * \text{blockCount}$ bytes of texture data to the LS address *outputBlocks*.

Notes

If any of the parameter alignment restrictions are violated, the low bits of the parameter will be ignored.

edgeDxtDecompress1

Decompresses texture data from the CELL_GCM_TEXTURE_COMPRESSED_DXT1 format.

Definition

```
#include <edge/dxt/edgedxt.h>
void edgeDxtDecompress1
(
    const void* inputBlocks,
    void* outputPixels,
    uint32_t rowStride,
    uint32_t blockCount
);
```

Arguments

<i>inputBlocks</i>	Pointer to where the compressed texture data will be read from Local Store. Must be 8-byte aligned.
<i>outputPixels</i>	Pointer to where the output pixel data will be written to Local Store. Must be 16-byte aligned.
<i>rowStride</i>	The number of bytes between each row of output pixel data. Must be 16-byte aligned.
<i>blockCount</i>	The number of 4x4 blocks of pixel data to decompress.

Description

This function decompresses texture data stored in the CELL_GCM_TEXTURE_COMPRESSED_DXT1 texture format within Local Store.

The color components of *outputPixels* are written in ARGB order.

Notes

If any of the parameter alignment restrictions are violated, the low bits of the parameter will be ignored.

edgeDxtDecompress3

Decompresses texture data from the CELL_GCM_TEXTURE_COMPRESSED_DXT23 format.

Definition

```
#include <edge/dxt/edgedxt.h>
void edgeDxtDecompress3
(
    const void* inputBlocks,
    void* outputPixels,
    uint32_t rowStride,
    uint32_t blockCount
);
```

Arguments

<i>inputBlocks</i>	Pointer to where the compressed texture data will be read from Local Store. Must be 16-byte aligned.
<i>outputPixels</i>	Pointer to where the output pixel data will be written to Local Store. Must be 16-byte aligned.
<i>rowStride</i>	The number of bytes between each row of output pixel data. Must be 16-byte aligned.
<i>blockCount</i>	The number of 4x4 blocks of pixel data to decompress.

Description

This function decompresses texture data stored in the CELL_GCM_TEXTURE_COMPRESSED_DXT23 texture format within Local Store.

The color components of *outputPixels* are written in ARGB order.

Notes

If any of the parameter alignment restrictions are violated, the low bits of the parameter will be ignored.

edgeDxtDecompress5

Decompresses texture data from the CELL_GCM_TEXTURE_COMPRESSED_DXT45 format.

Definition

```
#include <edge/dxt/edgedxt.h>
void edgeDxtDecompress5
(
    const void* inputBlocks,
    void* outputPixels,
    uint32_t rowStride,
    uint32_t blockCount
);
```

Arguments

<i>inputBlocks</i>	Pointer to where the compressed texture data will be read from Local Store. Must be 16-byte aligned.
<i>outputPixels</i>	Pointer to where the output pixel data will be written to Local Store. Must be 16-byte aligned.
<i>rowStride</i>	The number of bytes between each row of output pixel data. Must be 16-byte aligned.
<i>blockCount</i>	The number of 4x4 blocks of pixel data to decompress.

Description

This function decompresses texture data stored in the CELL_GCM_TEXTURE_COMPRESSED_DXT45 texture format within Local Store.

The color components of *outputPixels* are written in ARGB order.

Notes

If any of the parameter alignment restrictions are violated, the low bits of the parameter will be ignored.