



WIKIPEDIA
The Free Encyclopedia

[Main page](#)
[Contents](#)
[Featured content](#)
[Current events](#)
[Random article](#)
[Donate to Wikipedia](#)
[Wikipedia store](#)

Interaction

[Help](#)
[About Wikipedia](#)
[Community portal](#)
[Recent changes](#)
[Contact page](#)

Tools

[What links here](#)
[Related changes](#)
[Upload file](#)
[Special pages](#)
[Permanent link](#)

Not logged in [Talk](#) [Contributions](#) [Create account](#) [Log in](#)

Article [Talk](#)

Read [Edit](#) [View history](#)

S3 Texture Compression

From Wikipedia, the free encyclopedia

S3 Texture Compression (S3TC) (sometimes also called **DXTn** or **DXTC**) is a group of related [lossy texture compression algorithms](#) originally developed by lourcha et al. of [S3 Graphics, Ltd.](#)^[1] for use in their [Savage 3D computer graphics accelerator](#). The method of compression is strikingly similar to the previously published [Color Cell Compression](#),^[2] which is in turn an adaptation of [Block Truncation Coding](#) published in the late 1970s. Unlike some image compression algorithms (e.g. [JPEG](#)), S3TC's fixed-rate data compression coupled with the single memory access (cf. Color Cell Compression and some [VQ](#)-based schemes) made it well-suited for use in compressing [textures](#) in hardware-accelerated [3D computer graphics](#). Its subsequent inclusion in [Microsoft's DirectX 6.0](#) and [OpenGL 1.3](#) (via the `GL_EXT_texture_compression_s3tc` [extension](#)) led to widespread adoption of the technology among hardware and software makers. While S3 Graphics is no longer a competitor in the graphics accelerator market, license fees are still levied and collected for the use of S3TC technology, for example in [game consoles](#) and graphics cards. The wide use of S3TC has led to a [de facto](#) requirement for OpenGL drivers to support it, but the patent-encumbered status of S3TC presents a major obstacle to [open source](#) implementations,^[3] while implementation approaches which try to avoid the patented parts exist.^[4]

Contents

1 [Patent](#)

[Page information](#)

[Wikidata item](#)

[Cite this page](#)

Print/export

[Create a book](#)

[Download as PDF](#)

[Printable version](#)

Languages

[Deutsch](#)

[日本語](#)

[Português](#)

[Русский](#)

 [Edit links](#)

2 [Codecs](#)

3 [DXT1](#)

4 [DXT2 and DXT3](#)

5 [DXT4 and DXT5](#)

6 [S3TC Format Comparison](#)

7 [See also](#)

8 [References](#)

9 [External links](#)

Patent [\[edit\]](#)

The patent on S3 Texture Compression expires on October 2, 2017.^[5]

Codecs [\[edit\]](#)

There are five variations of the S3TC algorithm (named **DXT1** through **DXT5**, referring to the **FourCC** code assigned by Microsoft to each format), each designed for specific types of image data. All convert a 4×4 block of pixels to a 64-bit or 128-bit quantity, resulting in compression ratios of 6:1 with 24-bit **RGB** input data or 4:1 with 32-bit **RGBA** input data. S3TC is a **lossy** compression algorithm, resulting in image quality degradation, an effect which is minimized by the ability to increase texture resolutions while maintaining the same memory requirements. Hand-drawn cartoon-like images do not compress well, nor do **normal map** data, both of which usually generate **artifacts**. ATI's **3Dc** compression algorithm is a modification of DXT5 designed to overcome S3TC's shortcomings with regard to normal maps. **id Software** worked around the normalmap compression issues in **Doom 3** by moving the red component into the alpha channel before compression and moving it back during rendering in the **pixel shader**.^[6]

Like many modern image compression algorithms, S3TC only specifies the method used to decompress images, allowing implementers to design the compression algorithm to suit their specific needs, although the patent still covers compression algorithms. The nVidia GeForce 256 through to GeForce 4 cards also used 16-bit interpolation to render DXT1 textures, which resulted in banding when unpacking textures with color gradients. Again, this created an unfavorable impression of texture compression, not related to the fundamentals of the codec itself.

DXT1 [\[edit\]](#)

DXT1 (also known as Block Compression 1 or BC1) is the smallest variation of S3TC, storing 16 input pixels in 64 bits of output, consisting of two 16-bit RGB 5:6:5 color values c_0 and c_1 , and a 4x4 two bit lookup table.

If $c_0 > c_1$, then two other colors are calculated, such that $c_2 = \frac{2}{3}c_0 + \frac{1}{3}c_1$ and $c_3 = \frac{1}{3}c_0 + \frac{2}{3}c_1$. This mode operates similarly to mode 0xC0 of the [original Apple Video codec](#).^[7]

Otherwise, if $c_0 \leq c_1$, then $c_2 = \frac{1}{2}c_0 + \frac{1}{2}c_1$ and c_3 is transparent black corresponding to a [premultiplied alpha format](#).

The lookup table is then consulted to determine the color value for each pixel, with a value of 0 corresponding to c_0 and a value of 3 corresponding to c_3 .

DXT2 and DXT3 [\[edit\]](#)

DXT2 and DXT3 (collectively also known as Block Compression 2 or BC2) converts 16 input pixels (corresponding to a 4x4 pixel block) into 128 bits of output, consisting of 64 bits of [alpha channel](#) data (4 bits for each pixel) followed by 64 bits of color data, encoded the same way as DXT1 (with

the exception that the 4 color version of the DXT1 algorithm is always used instead of deciding which version to use based on the relative values of c_0 and c_1).

In DXT2, the color data is interpreted as being [premultiplied by alpha](#), in DXT3 it is interpreted as not having been premultiplied by alpha. Typically DXT2/3 are well suited to images with sharp alpha transitions, between translucent and opaque areas.

DXT4 and DXT5 [\[edit\]](#)

DXT4 and DXT5 (collectively also known as Block Compression 3 or BC3) converts 16 input pixels into 128 bits of output, consisting of 64 bits of alpha channel data (two 8 bit alpha values and a 4x4 3 bit lookup table) followed by 64 bits of color data (encoded the same way as DXT1).

If $\alpha_0 > \alpha_1$, then six other alpha values are calculated, such that $\alpha_2 = \frac{6\alpha_0 + 1\alpha_1}{7}$,
 $\alpha_3 = \frac{5\alpha_0 + 2\alpha_1}{7}$, $\alpha_4 = \frac{4\alpha_0 + 3\alpha_1}{7}$, $\alpha_5 = \frac{3\alpha_0 + 4\alpha_1}{7}$, $\alpha_6 = \frac{2\alpha_0 + 5\alpha_1}{7}$, and
 $\alpha_7 = \frac{1\alpha_0 + 6\alpha_1}{7}$.

Otherwise, if $\alpha_0 \leq \alpha_1$, four other alpha values are calculated such that $\alpha_2 = \frac{4\alpha_0 + 1\alpha_1}{5}$,
 $\alpha_3 = \frac{3\alpha_0 + 2\alpha_1}{5}$, $\alpha_4 = \frac{2\alpha_0 + 3\alpha_1}{5}$, and $\alpha_5 = \frac{1\alpha_0 + 4\alpha_1}{5}$ with $\alpha_6 = 0$ and $\alpha_7 = 255$.

The lookup table is then consulted to determine the alpha value for each pixel, with a value of 0 corresponding to α_0 and a value of 7 corresponding to α_7 . DXT4's color data is premultiplied by alpha, whereas DXT5's is not. Because DXT4/5 use an interpolated alpha scheme, they generally produce superior results for alpha (transparency) gradients than DXT2/3.







S3TC Format Comparison [\[edit\]](#)

FOURCC	DX 10 Name	Description	Alpha premultiplied?	Compression ratio	Texture Type
DXT1	BC1	1-bit Alpha / Opaque	Yes	6:1(for 24 bit source image)	Simple non-alpha
DXT2	BC2	Explicit alpha	Yes	4:1	Sharp alpha
DXT3	BC2	Explicit alpha	No	4:1	Sharp alpha
DXT4	BC3	Interpolated alpha	Yes	4:1	Gradient alpha
DXT5	BC3	Interpolated alpha	No	4:1	Gradient alpha












See also [\[edit\]](#)

- [Color Cell Compression](#)
- [S2TC](#), patentless workaround
- [3Dc](#)
- [FXT1](#)
- [DirectDraw Surface](#)
- [PVRTC](#)
- [ASTC](#)
- [ETC1](#)

References [\[edit\]](#)

1. [^](#) [US 5956431](#)  "Fixed-rate block-based image compression with inferred pixel values"
2. [^](#) ["1990 IEEE Color Cell Compression Paper"](#)  (PDF). [leeexplore.ieee.org](#). Retrieved 2012-01-25.
3. [^](#) ["S3TC situation on official DRI information page"](#) . [Dri.freedesktop.org](#). Retrieved 2012-01-25.
4. [^](#) [S2TC: A Possible Workaround For The S3TC Patent Situation](#)  on [phoronix](#)
5. [^](#) ["Can I play Morrowind with OpenMW on other platforms like the Raspberry Pi?"](#) .
6. [^](#) ["DOOM 3 Video Requirements"](#) . [Gamershell.com](#). Retrieved 2012-01-25.
7. [^](#) [Togni, Roberto, et al. "Apple RPZA"](#) . [MultimediaWiki](#).

External links [\[edit\]](#)

- [Microsoft Developer Network article on Block Compression in Direct3D 10](#) 
- [NVIDIA Texture Tools](#) 
- [AMD GPU Tools](#) 
- [squish](#) , an [MIT-licensed](#) S3TC compressor. The site also contains [an article](#)  giving an introduction to compression algorithms.
- [libtxc_dxtn](#) , a BSD licensed module for Mesa
- [Comparison between S3TC and FXT1 texture compression](#) 
- [The Truth about S3TC](#)  Note: This article used an early S3TC compression engine, not nVidia's or ATI's updated codecs.
- [Texture compression](#)  survey
- [crunch](#) , a [ZLIB-licensed](#) DXT1/5/N command line tool and compression library with a highly compressed intermediate format.
- [A fast, SSE2-enabled DXT1/5 compressor by Intel](#) 

Categories: [Lossy compression algorithms](#) | [Texture compression](#) | [3D computer graphics](#)

This page was last modified on 11 October 2016, at 07:41.

Text is available under the [Creative Commons Attribution-ShareAlike License](#); additional terms may apply. By using this site, you agree to the [Terms of Use](#) and [Privacy Policy](#). Wikipedia® is a registered trademark of the [Wikimedia Foundation, Inc.](#), a non-profit organization.

[Privacy policy](#) [About Wikipedia](#) [Disclaimers](#) [Contact Wikipedia](#) [Developers](#) [Cookie statement](#) [Mobile view](#)

