ILDA Image Data Transfer Format

ILDA Image Data Transfer Format

Written by:
Steve Heminover, Aura Technologies, Inc.
Patrick Murphy, Pangolin Laser Software

Prepared by:
Kelly and Frank Plughoff, Full Spectrum Lasers

ILDA Image Data Transfer Format

The International Laser Display Association has developed an "image data transfer" format, used to exchange frames between systems. You can obtain frames from any program with an ILDA conversion program, and transparently load them directly into any system that supports ILDA standard frames. Similarly, you can save frames in ILDA format, to sell or trade with users of other systems that read ILDA format.

The ILDA format is computer- and disk-independent. For example, it does not rely on or require files to be in PC format. To trade with another user who has ILDA files, you can either transfer the files by modem, or you can use disks in a format that both of you can read and write (such as PC 3.5" 1.44 MB).

The ILDA format was developed by ILDA's Technical Committee, under the direction of Steve Heminover of Aura Technologies. By its nature, the format had to support systems of differing abilities. There are some known shortcomings to the format. However, it works well in its stated goal to allow frame interchange from one system to another.

The most obvious shortcoming is in the area of colour. Originally, the ILDA format contained X, Y, Z and colour number information for each point. The colour number was arbitrary — colour #1 on one system might be black, while on another system it is white. For this reason, it was usually impossible to correctly transfer colours between different systems (at least, using the default colour palettes on each system).

To address this shortcoming, two proposals were made to ILDA. One was to have an "ILDA standard" colour palette. When loading or saving ILDA format, a system's default colours would be translated into the closest matching "ILDA standard" palette. This proposal was adopted, and is available separately from the Technical Committee.

The second proposal was to add a colour header to the ILDA format. The colour section would describe the RGB values used for each colour number. The ILDA-reading program could use this information to re-map colour numbers correctly. This proposal was adopted in June 1992, for Revision 004. However, to the best of our knowledge, no system reads or writes the colour header at the time of writing.

Introduction

Formatting

Items outdented like this are part of the standard.

> Items indented and in a smaller font like this, contain descriptive or explanatory material. They are useful in understanding and applying the standard, but they are not an official part of the standard.

Nomenclature and Structure

This document describes the official International Laser Display Association format for transferring single and/or multiple frames between systems.

> The purpose of the standard is transfer of graphic images — frames and animations — between systems. It is not optimised for space or speed, and it is not currently concerned with display issues such as point output rate.

The official name of this standard is "ILDA Image Data Transfer Format" or "ILDA Format". The official name for files conforming to this standard is "ILDA Format Files".

> There are other ILDA formats for areas such as hardware. Use the longer name of the standard on first reference.
>
> The first part of the ILDA Format incorporates the introduction. The second part covers the "ILDA Coordinates" section. It incorporates the two subsections "Coordinate Header" and "Coordinate Data".
>
> The third part covers the "ILDA Colour Tables" section. It incorporates the three subsections "Colour Table Header", "Colour Table Data" and "Colour Table Notes".
>
> For accurate communications, use the names given in quotes above when discussing the various parts of the ILDA Format.

Throughout this document, the word "must" is used in capitals to stress required conformance with the ILDA Format. The word "should" in capitals or lower case indicates suggested conformance.

Overview

> An ILDA file consists of sections with coordinate or colour information.
>
> There are currently three types of sections:
> Coordinate information for two-dimensional frames
> Coordinate information for three-dimensional frames
> Colour lookup table information
>
> Each section has two subsections, a header and data. The header subsection begins with the ASCII characters "ILDA". A format code identifies the section type: 0 for 2D, 1 for 3D, 2 for colour tables. Other information follows. The most important from a reading standpoint is information on the number of coordinates or colours. This is used by the reading program to determine how many times to read data. In format 0 and 1, there is one section (header plus data) for each frame. In format 2, there is one section for each defined colour table. Depending on the system, all frames may share the same colour table, each frame may have its own, or there may be some other implementation.

Binary vs. ASCII

The terms "binary 0" or "binary 1" refer to bit codes 0000 0000 and 0000 0001. They are used to avoid confusion with the ASCII characters "0" or "1".

Byte Order

Byte order is high bytes followed by low bytes. For example in a word, the two bytes 11111111 00000000 represent the decimal number 65280, not 255.

Physical Implementation

This standard does not specify any physical implementations, such as using IBM disk formats. It is a software-only format. It is up to the transferring parties to have the data readable by both systems. Where modem transfer is to take place, any CRC or checksum calculations are left up to the communications programs' protocol.

Upward Compatibility

A program which reads ILDA Format Files MUST be able to read all past revisions at time of programming. A program which writes ILDA Format Files SHOULD write the most current ILDA format at time of programming, but could write an earlier format if desired.

Skipping Unfamiliar Formats

A reading program MUST skip a section (ILDA header plus data) if the format byte in the sector is unfamiliar.

> For example, a program may understand format 0 (3D images) and format 1 (2D images). If it sees a byte indicating format 9 (currently undefined), it does not process further data except as necessary to find the ASCII letters "ILDA" indicating start of next header.

A reading program should calculate offsets (using "Total Points" or "Total Colours" data bytes) when skipping ahead, instead of blindly searching for the "ILDA" sequence. This is because "ILDA" may also occur within frame names or as an accidental sequence of data bytes.

ILDA Coordinates

This part of the standard describes both the 3D and 2D coordinate formats. (The third type of format is for colour tables.) Be sure to note and take action based on byte 8, the format code.

Coordinate Header

(ILDA header description — one for each frame)

Byte	*Description*	*Remarks*
1-4	"I", "L", "D", "A"	The ASCII letters ILDA, identifying an ILDA format header.
5-7	0, 0, 0	Three binary zero bytes (all bits zero). Not used but must be set to zero.
8	Format code	Binary 0 for 3D images or binary 1 for 2D. In 3D mode there are four words per point and in 2D mode there are three words per point.
9-16	Frame name	Eight ASCII characters with the name of this frame.
17-24	Company name	Eight ASCII characters with name of the company who created the frame.
25-26	Total points	Total number of points in this image in binary (1-65535). If the number of points is 0, then this is to be taken as the end of file header and no more data will follow this header.
27-28	Frame number	If the frame is part of a group such as an animation sequence, this represents the frame number. Counting begins with frame 0. Frame range is 0-65535.
29-30	Total frames	Total frames in this group or sequence. Range is 1-65535.
31	Scanner head	The scanner head or projector number that this frame or image is to be displayed on. Range is 0-255.
32	Future	Reserved for future use. Must be set to binary 0.

Coordinate Data

(Data description — one for each frame)

Byte	*Description*	*Remarks*
33-34	X coordinate	A 16-bit binary twos complement (signed) number. Extreme left is -32768; extreme right is +32767. (All directions stated using front projection.)
35-36	Y coordinate	A 16-bit binary twos complement (signed) number. Extreme bottom is -32768; extreme top is +32767.
37-38	Z coordinate	A 16-bit binary twos complement (signed) number. Extreme rear (away from viewer; behind screen) is -32768; extreme front (towards viewer; in front of screen) is +32767. These two bytes do not appear if the format code (byte 8) indicates 2D frames.
39-40	Status code	Bits 0-7 (lsb) indicate the point's colour number. This value is used as an index into a colour lookup table containing red, green and blue values. See ILDA Colour Lookup Table Header section for more information. Bits 8-13 are unassigned and should be set to 0 (reserved). Bit 14 is the blanking bit. If this is a 0, then the laser is on (draw). If this is a 1, then the laser is off (blank). Note that all systems must write this bit, even if a particular system uses only bits 0-7 for blanking/colour information. Bit 15 (msb) is the "last point" bit. This bit is set to 0 for all points except the last point. A 1 indicates end of image data. This was done for compatibility with certain existing systems; note that a zero in bytes 25-26 (Total Points) is the

		official end-of-file indication.
41-N	Next X coordinate	Repeat point format until last point has been written.
N+1	Next header	Next ILDA header follows. If the next header has a zero value for Total Points (bytes 25-26), then it is the last header in the file and the file can be closed.

ILDA Colour Tables

This part of the standard describes the colour table format. See the notes following for more information.

Colour Table Header

(ILDA header description — one for each colour table)

Byte	***Description***	***Remarks***
1-4	"I", "L", "D", "A"	The ASCII letters ILDA representing the organisation. Same as coordinate header.
5-7	0, 0, 0	Three zero bytes. Not used but must be set to zero (future use). Same as coordinate header.
8	Format code	Value "2" (0010 binary) for colour lookup table.
9-16	Palette name	Eight character ASCII name of palette. Should be identical to name used in coordinate header.
17-24	Company name	Eight character ASCII name of the company who created the palette. Should be identical to name used in coordinate header.
25-26	Total colours	Total number of RGB values defined in lookup table. For example, a digital RGB system with 1 bit each for R, G and B would specify eight colours. Although two bytes are used here, the number of colours must be between 2 and 255. This is because the colour Status Code is limited to 8 bits total. Single-colour systems should specify two colours, even if they do not support blanking. Colour #0 should have RGB values of 0-0-0, Colour #1 should have RGB values of 255-255-255. A value of 0 is reserved.
27-28	Palette number	A software program may have more than one palette lookup table. These bytes describe which lookup table is being defined. Counting begins with palette 0. Palette range is 0-65535. Systems with absolute colour can use one lookup table per frame. This allows any frame to contain up to 256 colours.
29-30	Future	Reserved for future use; must be zero.
31	Scanner head	This represents which head or scanner pair the lookup table(s) is being defined for. Range is 0-255.
32	Future	Reserved for future use; must be zero.

Colour Table Data

(Data description — one for each colour table)

Byte	*Description*	*Remarks*
33	Red #0 value	Intensity value of red for first colour in table (colour #0). Value ranges from 0 (off) to 255 (full on).
34	Green #0 value	Intensity value of green for colour #0.
35	Blue #0 value	Intensity value of blue for colour #0.
36	Red #1 value	Intensity value of red for second colour in table (colour #1).
37	Green #1 value	Intensity value of green for colour #1.
38	Blue #1 value	Intensity value of blue for colour #1.
39+		Repeat red, green and blue for each entry. Total entries must equal value in colour header bytes 25-26.

Colour Table Notes

Introduction

Coordinate information — a point's location — is straightforward. The use of colour lookup tables is not. There are many different ways of using blanking and colour. There are many different ways of interpreting the colour information in the ILDA Coordinates section.

Because systems and implementations vary, the Colour Table Notes go into detail. They help standardise the transfer of colour information even across very different systems.

Nomenclature

Special terms used in this document are defined as follows. These definitions may be more specific than those in the ILDA Laser Glossary.

Blanking

Control of the laser's brightness on a point-by-point basis, either by turning it on and off or by varying its intensity. In this document refers to both digital and analogue techniques.

Digital Blanking

The beam can only be turned on or off.

Analogue Blanking

Continuously variable control of the laser brightness. "Intensity" is often used as a synonym.

Digital RGB

Mixing the three primary colours of light by turning them on or off. Can produce 8 total colours: red, green, blue, yellow (R+G), magenta (R+B), cyan (G+B) and white (R+G+B), with black the eighth "colour".

Analogue RGB

Mixing the three primaries using intensity control. Many systems have 256 levels of control over each colour, providing 16,777,216 possible combinations of red, blue and green levels.

Spectrum Colour

A single signal causes colour change. Usually done by moving a prism to select different wavelengths; hence the name.

Colour-blanking

Describes systems which use only the colour devices to blank. Many analogue RGB and spectrum colour systems use this technique.

Separate-blanking

The opposite of colour-blanking. A system which has a blanking device controlled separately from any colour device. By definition also incorporates single colour, blanking-only systems.

Use of the Colour Header

The reading program should assume a single-colour image in the absence of any Colour Table Header. Reading programs MUST read in both older, non-colour-header files, and newer files with colour headers.

An image file does not have to contain a colour header.

If a colour header is found as part of an ILDA Format File (along with coordinate headers and data), the colour header may appear anywhere in the file — before, in the middle of, or after sectors with coordinate headers and data.

It is also possible to make "palette files" that contain only a colour header.

The reading program MUST be able to read colour Table Headers and Data anywhere in the file — even if they are the only elements of the file.

It is strongly suggested that the colour header be the first header in a file, but this is not a requirement. In most systems, there is only one, or a few, colour lookup tables. When a new table is loaded, all existing colours change to those in the new lookup table. This means that the last colour header encountered takes precedence over all prior colour headers that have the same lookup table number (byte 26).

There may be other systems where every frame can have its own palette. (Absolute colour systems would be handled like this.) To provide for these cases, the following rule applies: the colour header defines a palette for the following frames. If a new colour header appears, frames appearing afterwards

take on the new palette values. This is why the colour header should be the first header in a file.

The writing program should write a Colour Table Header and Data before Coordinate Headers and Data. The colour header defines a palette for all frames which follow until the next colour header (for multiple-palette systems).

Colour, Intensity and Blanking

The following rules are defined so that colour usage on separate-blanking and colour-blanking systems is consistent.

All writing programs MUST write bit 14 of the Status Code. This blanking bit is a 0 if laser is on (draw) and a 1 if laser is off (blanked).

For separate-blanking systems, then this bit simply reflects the blanking status. For colour-blanking systems, then this bit MUST be 0 if any of the RGB values is greater than 0, and MUST be a 1 if the RGB values are all 0.

If a system does not use 8-bit colour resolution, the colour resolution MUST be mapped onto 8 bits.

For example, if a system has 2-bit resolution (four different colour levels), the level is multiplied to fit full scale 0-225. Two-bit "0" remains 8-bit "0", 2-bit "2" becomes "85", "2" becomes "170" and "3" becomes "255".

Even if a system does not use any colour or blanking, it is suggested to write a Colour Table Header. For compatibility with the proposed ILDA Standard Colour Palette, specify just two colours in the table — Colour 0 with RGB values of 0, and Colour 1 with RGB values of 255 — and write all points using Colour 1.

Reading programs have a number of ways to derive blanking, intensity and colour information. Some choices are left up to the programmer to allow some flexibility. The following sections discuss these choices for different types of systems.

Separate-blanking Systems

Digital Blanking

Obtain blanking information either from the blanking bit, or derive it from the RGB information.

For example, a programmer could decide to blank any points whose combined RGB values are less than 10% of full scale. This option ensures that very dim lines are not drawn by a digital (on/off) system.

Analogue Blanking with Digital RGB

Derive intensity information from the RGB information. Convert the analogue RGB levels into digital RGB levels (0 or 255).

It is up to the reading program to determine the mapping algorithm that translates analogue RGB into the desired information. For example, the intensity could be merely the average of the combined RGB values.

Colour-blanking Systems

Analogue RGB

Reading programs using analogue RGB should use the RGB information directly.

Backwards Compatibility

Colour-blanking Systems

If bit 14 indicates blanked, use a colour number with RGB registers all set to zero. This ensures that blanking takes precedence over the colour number in Status Code bits 0-7.

Conversely, if bit 14 indicates visible, check to be sure the Status Code colour number in bits 0-7 has combined RGB values greater than 0. This ensures that visible points are indeed visible.

> Reading programs which read image files without colour headers have no idea which colour numbers get which RGB values. The only absolute colour information is the Status Code blanking bit.
>
> This presents a problem for colour-blanking systems. It is possible that the Status Code colour number bits (0-7) will not match the blanking bit (bit 14). That is, because there is no colour header, a blanked point could be assigned a colour which would make it visible on these systems. The system above solves this dilemma.

Defining RGB

Red, green and blue wavelengths are assumed to be colour-balanced, although exact wavelengths are not specified. If your system uses non-standard RGB wavelengths, you MUST correct as best you can when reading and writing.

> The system used by ILDA provides eight bits each of red, green and blue information. This makes for 16.7 million potential colours. These RGB values are defined in absolute terms; that is, they are not necessarily the specific RGB signals which may be sent to your particular colour system.
>
> The ILDA standard assumes colour-balanced RGB wavelengths similar to those used in high-quality computer monitors. Those wavelengths are not specified in the standard, so exact colour matching will not be possible at this time. However, you must read and write colour-balanced RGB levels even if your system does not use those internally. (You will need to use a second lookup table to convert from your projector's RGB signals to "absolute" RGB.)