

JEDMICS C4 COMPRESSED IMAGE FILE FORMAT  
TECHNICAL SPECIFICATION

FOR THE

JOINT ENGINEERING DATA MANAGEMENT INFORMATION AND CONTROL  
SYSTEM

(JEDMICS)

APRIL 2002

PREPARED BY:

NORTHROP GRUMMAN INFORMATION TECHNOLOGY

7575 COLSHIRE DRIVE

MCLEAN, VIRGINIA 22102

CAGE CODE: 1V4D7

DISTRIBUTION STATEMENT A

APPROVED FOR PUBLIC RELEASE: DISTRIBUTION IS UNLIMITED

## Revision History

Revision	Date	By	Comment
1.0	08-Apr-02	dfk	<p>The significant changes from Appendix B of MIL-PRF-28002C are:</p> <ul style="list-style-type: none"> <li>• Preview size is explicitly limited to 0 or 6 tiles</li> <li>• The ordering of the C4 Tile Index is explicitly stated</li> <li>• Changed the definition of the Negative Compression Flag in the C4 Tile Index to specify that any non-zero value sets the flag.</li> <li>• Explicitly specified that all reserved / not used fields in the C4 Header must be null padded to the specified size.</li> <li>• Explicitly specified that the C4 Tile Index size of a tile with negative compress is 32768.</li> <li>• Removed the QA Flag field from the C4 Header</li> <li>• Reformatted to make the text easier to follow</li> </ul>

## Table Of Contents

1.0	Scope.....	4
1.1.	Identification.....	4
1.2.	Audience .....	4
1.3.	Bibliography .....	4
1.4.	Definition of Terms .....	4
2.0	Requirements .....	6
2.1.	Overview.....	6
2.2.	Structure Formats .....	6
2.3.	Implementation Notes .....	9

## 1.0 Scope

### 1.1 Identification

This document provides the specification of the JEDMICS C4 image format as currently implemented with the JEDMICS 3.2 baseline.

Note that the information in this document supercedes the information provided in Appendix B of MIL-PRF-28002C, dated 30 September 1997.

### 1.2 Audience

Programmers and OEM manufacturers utilizing the JEDMICS C4 image format.

### 1.3 Bibliography

- “Facsimile Coding Schemes and Coding Control Functions for Group 4 Facsimile Apparatus”, *Recommendation T.6, Volume VII, Fascicle VII.3, Terminal Equipment and Protocols for Telematic Services*, The International Telegraph and Telephone Consultative Committee (CCITT), Geneva, Switzerland, 1985.
- “IMageCenter User’s Guide”, Image Machines Corporation, Herndon, 1991.
- “TIFF Revision 6.0”, Aldus Corporation, Seattle, 1992.

### 1.4 Definition of Terms

<b>aperture card</b>	A specialized version of the 80-column Hollerith card that has a microfilm copy of a graphic image mounted in a rectangular hole located in the area between columns 53 and 77. Hollerith text can be punched into the remaining columns of the card.
<b>big-endian</b>	The practice of storing numeric data starting with the most significant byte in the lowest memory location associated with the data. Each successive byte stored has a higher memory address and a higher mathematical significance. Thus, a 16-bit integer having the value 0x1234 would be stored as 0x12 0x34 in memory, and a 32-bit integer having the value 0x12345678 would be stored as 0x12 0x34 0x56 0x78. IBM mainframe processors and Motorola microprocessors employ this practice.
<b>bi-level</b>	A graphic image encoding scheme that only distinguishes between black pixels and white pixels.
<b>C4 header</b>	The first 128 bytes of a C4 file.
<b>C4 tile index</b>	The portion of C4 file directly following the header and that contains an entry describing each tile of the image.
<b>CCITT Group 4</b>	A facsimile-compatible data compression technique defined by the International Telegraph and Telephone Consultative Committee (CCITT). This scheme is defined in Recommendation T.6.

<b>compressed tile</b>	A tile which, due to the application of CCITT Group 4 data compression, can be described in less than 32768 bytes.
<b>dpi</b>	A representation of image resolution (dots per inch).
<b>Huffman encoding</b>	A file compression technique that represents data elements with variable length compression codes, shorter codes being used for more commonly occurring data elements and longer codes for less frequent elements.
<b>Intel byte order</b>	See <i>little-endian</i> .
<b>little-endian</b>	The practice of storing numeric data starting with the least significant byte in the lowest memory location associated with the data. Each successive byte stored has a higher memory address and a higher mathematical significance. Thus, a 16-bit integer having the value 0x1234 would be stored as 0x34 0x12 in memory. A 32-bit integer having the value 0x12345678 would be stored in memory as 0x78 0x56 0x34 0x12. Intel and Digital Equipment Corporation processors employ this practice.
<b>negative compression</b>	A situation in which data compression results in an item larger in size than its uncompressed counterpart.
<b>network byte order</b>	See <i>big-endian</i> .
<b>non-adaptive</b>	Huffman codes are frequently adapted to a sampling of the data being compressed. By comparison CCITT Group 4 compression uses a standard set of Huffman codes for all image data. This approach is called non-adaptive.
<b>overview</b>	See <i>preview</i> .
<b>pixel</b>	The smallest unit of an image that can have color and intensity. Stands for <i>picture element</i> and is also called <i>pel</i> .
<b>pixel tile</b>	See <i>tile</i> .
<b>preview</b>	A subimage that has been constructed to provide a rendition of the full image scaled to a screen resolution of 1024 by 1536 pixels (6 tiles).
<b>raster image</b>	A model of graphic image representation in which the image is expressed as a matrix of pixels similar to the arrangement of the phosphor dots on the screen of a cathode ray tube.
<b>resolution</b>	An expression of the pixel density of an image.
<b>row-major order</b>	An algorithm for addressing a one dimensional array as a multi-dimensional array. In the case of a two-dimensional array with a row size of $n$ , the first $n$ elements of underlying one-dimensional array constitute the first row of the two-dimensional array, the second $n$ elements the second row, etc.

**tile** A portion of an image file that represents an area 512 x 512 pixels (262,144 pixels or 32768 bytes). The pixels within a tile are arranged in row-major order.

## 2.0 Requirements

### 2.1 Overview

The JEDMICS C4 compressed raster image file format (referred to in the rest of the document simply as a C4 image) was developed by Image Machines Corporation of Herndon, VA. It is very similar in structure to Aldus Corporation's TIFF 6.0 tiled file format and Image Machine Corporation's TG4 file format. These three file formats differ only in the structure of the header and tile indices. All three file types use CCITT Group 4 encoding and the structure of the compressed tiles is the same for all three.

Each C4 image file is capable of storing a single black and white image (it has no capability to store color or gray scale information.).

When originally developed, there was a need to provide a method to present a scaled, full page display of the image contained within the C4 image file quickly. The PC hardware at the time was not capable of scaling the main compressed image data within the C4 image file quickly, so the option for including a small (six tile), pre-rendered overview image of the main compressed image was included in the C4 image file specification. Modern PC hardware does not have a problem scaling the main compressed image for display, so the generation of a preview image is no longer needed.

In the following file component description, numeric fields have one of three formats:

- **single** byte unsigned integer
- multi-byte unsigned integer with **Intel** (little-endian) byte order
- multi-byte unsigned integer with **network** (big-endian) byte order

### 2.2 Structure Formats

The JEDMICS C4 compressed raster image file consists of the following components in the listed order:

1. C4 header
2. C4 tile index (one entry for each 512 by 512 pixel tile in the compressed image data)
3. Compressed image data (composed of one or more CCITT Group 4 compressed 512 by 512 pixel tiles)
4. Preview tile index (one entry for each 512 by 512 pixel tile in the preview compressed image data)
5. Preview compressed image data (composed of either zero or six compressed 512 by 512 pixel tiles representing a pre-rendered scaled view of the compressed image data)

## 2.2.1. C4 header

### 2.2.1.1. Header Structure Format

The description of the following C4 header fields is from the perspective of the PRC Digital Image Library (DIL). The fact that some fields are unused is an indication that these fields have lost significance in the DIL since this file format was originally designed. The reserved areas in the header were included to allow for future expansion but also to force word and double word fields to word boundaries, since some C compilers do not have a pragma for specifying packed data.

Note that gaps are not allowed in the C4 Header. All fields listed as 'null fill' should be null padded as necessary to the specified byte size.

Field name	Data format	Byte size	Byte offset	Description
index offset	Intel	4	0	Byte offset from the beginning of the file to the tile indices.
line height	Intel	2	4	Height of the image in pixels.
byte width	Intel	2	6	Width of the image in bytes.
data offset	network	4	8	Byte offset from the beginning of the file to the compressed tiles.
number of tiles	single	1	12	Number of 512 x 512 pixel tiles in image: if number of tiles is > 252, then value is 0, else value is number of tiles.
reserved	null fill	1	13	Forces the next field to a word boundary.
preview index offset	Intel	4	14	Offset from the beginning of the file to the preview tile index. If set to 0, then there is no preview data contained in the file.
preview height	Intel	2	18	Height of the preview in pixels.
preview width	Intel	2	20	Width of the preview in bytes.
preview data offset	network	4	22	Offset from the beginning of the file to the compressed preview tiles.
preview number of tiles	single	1	26	Number of 512 x 512 pixel tiles in the preview. Either 0 or 6.
reserved	null fill	1	27	Forces the next field to a word boundary.
file type	network	2	28	unused (set to zero)
QA flag	network	4	30	unused (set to zero)
reserved	null fill	2	34	unused
format	single	1	36	Data format code: if the number of tiles is > 252, then the value is 6, else the value is 4.
index spacing	null fill	1	37	unused

Field name	Data format	Byte size	Byte offset	Description
dpi	network	2	38	Resolution (dots per inch) at which image was scanned (up to 400 dpi). A value of 0 or a value of greater than 400 indicates 200 dpi. As a general rule, 200 dpi should be used for maximum compatibility with legacy applications.
pixel polarity	null fill	1	40	unused
Hollerith	ASCII	80	41	Aperture card Hollerith data converted to an 80 character, null padded ASCII string. May be used to store other data, if the image was not scanned from an aperture card.
reserved	null fill	7	121	Forces the tile index to start on a sector boundary.

#### 2.2.1.2. Header Error Conditions

A C4 file whose header exhibits any of the following conditions is considered invalid:

- *index offset* not equal to the size of the header (128).
- *format* not equal to either 4 or 6.
- *format* is equal to 4 and the number of tiles (as calculated from *line height* and *byte width*) is greater than 252.
- *format* is equal to 6 and the *number of tiles* is not 0.
- *data offset* not equal to the size of the C4 header plus the *number of tiles* times the size in bytes of a tile index (4).

#### 2.2.2. C4 Tile Index

If the *number of tiles* in the compressed image data is greater than 252, then the order of the entries in the C4 Tile Index must match the order of the tiles in the compressed image data (i.e. row-major ordering). This is because the *tile number* value is ignored for a situation where the *number of tiles* is greater than 252.

If the *number of tiles* in the compressed image data is less than or equal to 252, then the ordering of the entries in the C4 Tile Index does not need to match the ordering of the tiles in the compressed image data. This is because the *tile number* entry in the C4 Tile Index maps to the corresponding tile in the compressed image data.

Tiles in the compressed image data are numbered in row-major order, i.e., tile 0 is at the image's top left corner, tile 1 is to its right, and so forth to the end of the row. The number of tiles in an image is always computable directly from the image's line height and byte width:

```

tile height = (uint) floor((line height + 511) / 512)
tile width = (uint) floor((byte width * 8) + 511) / 512)
tile count = tile height * tile width
tile row number = (uint) floor(tile number / tile width)
tile column number = (tile number % tile width)
highest tile number = total number of tiles - 1

```

Field name	Data format	Byte size	Byte offset	Description
tile number	single	1	0	Zero-based offset (with respect to the beginning of the compress image data of the image file) of the tile in the image. If the number of tiles is > 252, then this value is ignored (and should be set to zero), else the value is the zero-based tile number (i.e. the first tile on the first row is 0, the second tile on the first row is 1, and so on).
negative compression flag	single	1	1	If the compress image data tile exhibits a negative compression (i.e. the CCITT Group 4 compressed tile is larger than the raw pixel tile), then the value is set to a non-zero value (1 is recommended), else the value is 0. A tile that would exhibit negative compression is stored as a raw pixel tile.
data size	Intel	2	2	Size of the compressed image data tile in bytes. If the tile is stored in raw mode (i.e. <i>the negative compression flag</i> is set for the tile), then this field should be set to 32768.

### 2.2.3. Compressed Image Data

A C4 image is segmented and stored as a collection of 512 by 512 pixel tiles, each of which is individually compressed, using CCITT Group 4 encoding, as though it were an independent image. If the tile exhibits a negative compression characteristic (i.e. the CCITT Group 4 compressed tile is larger than the raw pixel tile), then tile is stored as a raw pixel tile and the *negative compression flag* for the tile is set in the C4 Tile Index.

Since a C4 image width is expressed in bytes and the actual image width might not be modulo 8, edge tiles sometimes contain unused space. The unused parts of rightmost and bottommost edge tiles are ignored and are padded with 0 values as necessary to achieve best possible compression

### 2.2.4. Preview Tile Index (optional)

The preview tile index has the same structure as the C4 tile index.

### 2.2.5. Preview Compressed Image Data (optional)

The preview compressed image data has the same basic structure as the compressed image data.

The preview compressed image consists of either zero or six tiles (logically arranged as three rows of two tiles each).

## 2.3. Implementation Notes

### 2.3.1. Tile number

Each tile index structure contains a *tile number* field that identifies the position of that tile in the logical image. Tile number '0' indicates the tile in the upper left-hand corner of the image. Tile numbers proceed across the image in row-major order.

If the image file has 252 tiles or less, the *format* field of the header contains 4 and the *tile number* field of each tile index entry is significant. In this case, the tile index entries need not be in any specific order. (Historically, the limit of 252 was forced by the aperture card technology available at the time the C4 image format was developed.)

If the number of tiles in the image exceeds 252, the *format* field of the header contains 6 and the *tile number* field of each tile index is set to zero, since the *tile number* field is not large enough to house a meaningful tile number. In this case the tile index entries must be arranged in row-major order.

### **2.3.2. Resolution**

If the *dpi* field is less than or equal to 0 or greater than 400, image processing software should assume a value of 200 dpi. This value applies to both horizontal and vertical resolution, since C4 format assumes that the image is stored at a 1:1 aspect ratio.

As an historical note, all legacy JEDMICS hard copy scanning hardware produced 200 dpi C4 images.

### **2.3.3. Negative Compression**

The *negative compression flag* in the tile index structure indicates a case where compression of the given tile would have resulted in a larger data size than the original tile. In this event, the tile is not compressed and the *negative compression flag* is set. The *data size* of the tile in this case will be 32768.

### **2.3.4. Preview**

Applying a scaling algorithm to the full image and compressing it produces the preview. The goal of the scaling algorithm is to reduce the size of the image to fit onto a screen of 1024 x 1536 pixels, (i.e. six tiles); the scaling factor thus varies with the image size.

For historical reasons, the preview compressed data size is either zero tiles (i.e. no preview) or exactly six tiles.

The original purpose of the preview was to provide a small version of the main image to be used at lower zoom levels thus relieving slower processors (e.g., Intel 80286) from some resource intensive scaling. Applications written for more powerful processors need not make use of the preview. As a rule, use of the preview feature of the C4 image is discouraged unless there is a need to interface with old legacy applications and/or hardware.

### **2.3.5. Pixel Polarity**

A pixel value of 1 indicates black, and 0 white.

### **2.3.6. Features of CCITT Group 4 Compression**

- Each tile is a separate image 512 by 512 pixels in size.
- Each scan line within a tile is a multiple of 8 pixel bytes wide.
- Each tile follows an imaginary row of white pixels.

- Uses left to right bit ordering (i.e., the first bit of a tile's first scan line's compressed data is stored in the leftmost bit of the first byte of the compressed tile data).
- A scan line is encoded as one or more runs of black pixels and/or white pixels using non-adaptive Huffman codes.
- A run is composed of zero or more Makeup code words and one Terminating code word. Pixel runs less than 64 can be encoded with a single Terminating code word and runs of 64 to 2623 pixels are encoded with a single Makeup code and a Terminating code.
- A color transition within a scan line can be represented as being a certain number of pixels from a color transition on the current line or previous line.
- A 24-bit End-of-Facsimile Block (EOFB) follows the encoding of the last line of the pixel array immediately, in place of any additional line encoding.

000000000001000000000001

Enough 0-bit padding to fit the entire stream into a sequence of 8-bit bytes immediately follows the EOFB sequence.