ZOIA Patch File binary Format Technical Document

The ZOIA is a musical instrument effect pedal created by Empress Effects Inc. The ZOIA is a very complex device in which audio effects are created by adding virtual modules and connecting them together to manipulate the sounds. Such configurations are called patches. Furthermore, the ZOIA has a slot for a micro-SD flash card where patch files can be stored and transferred to and from the ZOIA's active list of patches.

This document is a walkthrough of what is known and speculated about the ZOIA patch files. The goal is to help people who would like to create software that could read and display the content of patch files.
This document is part of the [meanmedianmoge/zoia_lib] GitHub repository and has been created and maintained by the contributors with the help of Steeve Bragg of Empress Effects Inc.

Applicability

There are no guarantees that the information contained in this document is exact. We tried to verify it to the best or our abilities but we make no promises.

The information herein has been verified against version 2.50 and 2.70 of ZOIA's firmware.
See the appendix for the revision history of this document.

Also pay attention to the limitation mentioned in the ZOIA Specifics section below concerning module types configuration.

General Definitions and Conventions

- Numeric values without any special notation are in base 10 (as opposed to hexadecimal).
- Hexadecimal values are prefixed with 0x, ex.: 0x3e or 0x2.
- When the text refers to a particular section or field in the patch file, the corresponding name is enclosed in square brackets ([]) for clarity.
- The «**n/a**» mention means «not available» or «unknown» or «irrelevant» depending on the situation.
- The term «**index**» is a numerical value that identifies an item in a list of objects by its position. The first item is zero, the second is one, etc.

ZOIA Specifics

About Modules

To be able to decode a patch file, it's important to understand some things about modules how we name some things.

Module Type vs **Module Instance** – ZOIA provides a bunch of different module **types** (ex: flanger, lfo, etc.) that are used to create patches. In a patch, we insert instances of module types.

To be able to decode the content of a patch file, we need a lot of information about each available module type. For example, in a patch file, we find modules with an array of values for the options selected when the module instance was created. We need a reference document that helps us understand what means a value of 2 for option index 0 in a particular module data in a patch file. Another example is the connections between modules. In the patch file, connections are defined using a combination of module instance index and source and destination blocks numbers. Again, we need some document that tells us that a given block number refers to the "audio input left" or "audio input right" or some other part of the module instance.

Modules types and instances alike are composed of what we call «**blocks**» with which we interact when creating patches. Blocks correspond to buttons on the ZOIA grid. Blocks can be seen as inputs and outputs for the module. Blocks can be used to receive or send audio data while others can be used to input or output CV values. Regarding blocks, it is important to understand that:

- For many module types, some blocks might appear or disappear from the ZOIA grid depending on the options we choose when editing the module instance. We consider that some options simply **hide** some blocks. The blocks are still there but hidden.

Furthermore, the description of a module type might change as firmware changes can provide new features for any existing module type. Since we want to be able to read patch file that might have been saved with any older firmware, we would in theory need the description of the module type for each of the known versions that ever lived. In this text, we use the term «**blocks configuration version**» to refer to all the information that describes one specific version of a module type.

So here is a description of the data structure we need in order to understand the content of a patch file.

We need to define each Module Type that ZOIA supports. Each module type is described with the following data:

1. The unique numerical **Id** (identification) for the module type. The list in appendix A gives that list.
2. The **name** (text) for the module type. Ex.: «LFO», or «flanger», etc.
3. A category name can also be attached to a module type as the module types are access by categories in the ZOIA but it is of no real help in understanding a patch file.
4. A list of **Blocks Configuration Versions** for each known (or supported) version of the module type.

A **Blocks Configuration Version** is described with the following data:

1. A unique **version** identifier. It's an unsigned 32 integer value. Each blocks configuration for a given module type has a unique version number.
2. It might also be interesting to associated a ZOIA firmware version string to identify the firmware relating to this configuration of blocks.
3. A list of all the **options** supported by the module type (under that version). For each option we need:
    a. The index of the option's value in the data we find in the patch file for module of that type. Its like a unique Id number for the option.
    b. The option's name.

   c. The values that the option can take. This can have one of two forms:
      i. A list of choices (strings) with corresponding internal numerical values. Ex: value 0 for «off» and value 1 for «on».
      ii. A valid range of numbers. Ex.: for a midi channel options, the range of values is the numbers from 1 to 16.
4. A list of all the **blocks** supported by the module type (under that version). Note For each block we need:
   a. The block number. A unique integer value that identifies a specific block for the ZOIA. This is what identifies specific blocks in connections.
   b. The block name. The text that is seen on the ZOIA display when selecting that block.
   c. The type of data associated with the block (either CV data or Audio data).
   d. The direction of the data flow for this block (either input or output).
   e. The units in which the values are represented. Applicable to CV input blocks only. Note: this remains much under construction. No real details are available at the time of this writing.
   f. The conditions under which this block is hidden. This is optional. It describes the combination of option and associated values that causes this block to be hidden.

We already established that a fair amount of data is required to describe all the available module types with the associated blocks and options, etc. At the time of this writing, such a file is being constructed and will «probably» be made available in the form of a Json configuration file at a later time.

Patch File Format

Patches are all 32 KB in size. Unused portions are zeroed out.

Some data format conventions:

- Unsigned multiple-bytes numeric values are stored least-significant-byte-first. For example, value 123456 (0x01E240) would be stored as the following sequence of values: 0x40, 0xE2, 0x01, 0x00.

- Texts like patch names, modules names, etc., are usually of fixed size and are stored one byte per character and are coded as ASCII characters. Since the ZOIA interface only allow for the following characters, we don't expect to find anything else except for values 0x00 when the text is shorter than the maximum allowed size:

  - lowercase letters a-z (0x61 – 0x7A)
  - Uppercase letters A-Z (0x41 – 0x5A)
  - Numbers 0-9 (0x30 – 0x39)
  - Special characters: space (0x20), dash (0x2D) and dot (0x2E).

- The «field offset» and «field length» values in the following sections are stated as a number of bytes unless otherwise noted.

- All offset values are zero-relative meaning that the offset of the first byte in a group is zero (0).

- In the tables below, the «field offset» values are relative to the start of the section or the sub section that they describe. Meaning that the first byte in the section is zero even if the section might start at position 1000 in the patch file.

Patch file are composed of sections as described in the following table. Each section is defined in more details in the following sub sections of the document.:

Position	Name	Description
1	Patch Header	This section defines the size of the patch definition and the patch name.
2	Modules	This section contains data for each module used in the patch.
3	Connections	This section describes the connections between modules inputs and outputs.
4	Pages Names	This section contains the names assigned to page in the patch, if any. NOTE: This section can be empty (contains no page name) if none of pages were given names when the patch was created. This means that we have to look at the [Page Number] field in the modules list to figure out the actual number of pages in the patch.
5	Starred Elements	This section defines information on starred elements inside the patch. ZOIA stars can be applied either to individual module's parameters or to connections.
6	Modules Colors	Since firmware version 1.10 and above, this section defines the (extended) color assigned to each module. The number of colors defined and the order is the same as in the [Modules] section.

Here is a hexadecimal representation of binary patch file which includes the modules Audio in, Audio out with two connections established between those two modules:

```
        [--Preset size----][--------------------Patch name ----------------------------------------][--Num mods--][----------------------------------------Module #1 ----------------------------------
0000    39 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00|02 00 00 00|0e 00 00 00|01 00 00 00|00 00 00 00|00 00 00 00|03 00
        ----------------------------------------------------(module info continued)-----------------------------{----------------------------------mod name-----------------------}][------------------
002a    00 00 03 00|00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00|00 00 0e 00|00 00
        ------------------------------------------------------------------------------Module #2-----------------------------------------------------------------------------------------------------------------
0054    02 00 00 00|00 00 00 00|00 00 00 00|04 00 00 00|07 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00|00 00
        ----------------------------------------------------][--#connects---][----------------------------------Connection #1-----------------------------------------------][----------------------------
007e    00 00 00 00|00 00 00 00|00 00 02 00|00 00 00 00|00 00 00 00|00 00 01 00|00 00 01 00|00 00 10 27|00 00 00 00|00 00 01 00|00 00
        ------------Connection #2-----------------------------------][---num pages--][--------------------Page #1 name-----------------------------------------][-----------------------Page #2 name-------
00a8    01 00 00 00|00 00 00 00|10 27 00 00|02 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00|00 00 00 00|70 61 72 61|6d 73 00 00|00 00
        --------------------------------][-# star params-][-Mod #1 color-][-Mod #2 color]
00d2    00 00 00 00|00 00 00 00|00 00 08 00|00 00 0f 00|00 00
```

Patch Header section

Field Offset	Field Length	Field Name	Description
0	4	Patch Size (or preset size)	Unsigned long integer – the size of the patch file as a number of 4 bytes «chunks», including of itself. Multiply this value by 4 to get the total number of bytes.
4	16	Patch Name	Text – the patch name.

Modules section

The modules section contains data for each module used in the patch. Because each module takes up a variable number of bytes in the patch file, it's easier to describe this section using two tables. The following table defines the general format for the whole [Modules] section while the [Module definition] table defines the fields for one module in the list.

Field Offset	Field Length	Field Name	Description
0	4	Number of modules	Unsigned long integer – the number of [Module definition] that follows.
4	variable	List of [Module definition]	A list of [Module definition]. The number of modules is given by the previous field. The format of an individual module in this list is described in the table below.

Module definition

The following table describes the format of each module in the list of modules. The field offset is relative to the start of the module in the list.

Field Offset	Field Length	Field Name	Description
0	4	Module Size	Unsigned long integer – the size of the module definition as a number of 4 bytes «chunks», including of itself. Multiply this value by 4 to get the total number of bytes.
4	4	Module Type ID	Unsigned long integer – the module type ID. See appendix A for more details.
8	4	unknown	Unsigned long integer – this field has a value of zero 99% of the time. Otherwise, a value 1 has been observed from time to time. More work is required here.
12	4	Page number	Unsigned long integer – the page number where the module is located. Starts at zero for the first page.
16	4	Old Color	Unsigned long integer – the old color number that would be used to display this module on a ZOIA with pre 1.10 firmware. Colors are defined in appendix B. For patches created with firmware 1.10 and beyond, the color used to display the module is defined in the [Modules Colors] section. In this case, the value in this field is the closest match old color. Details can be found in appendix B.
20	4	Grid Position	Unsigned long integer – The position of the first (leftmost) module cell (led) on the page. It's a number from 0 to 39. From left to right, the first row are numbers 0-7, second row are 8-15, etc.
24	4	Number of User Parameters	Unsigned long integer – ?? What exactly we mean by «user» ?? Most of the times (but not always) this is the number of values in the [Additional Options] field.

28	4	Module Version	Unsigned long integer – This would be more appropriately called a «module type version». It refers to a specific version of a module type that was current in the ZOIA when the patch was saved. The module type configuration (options, blocks numbers, ...) relating to that version must be known in order to decode that module's data correctly.
32-39	1	Module Option 0 to Option 7	Unsigned byte integer – Values for options 0 to 7 as set by the user when creating or editing the module. These are all single-byte values so they can range from 0 to 255. There is room for 8 option's value but the number of values that are really used depends on the module type and version. Used options are always at the beginning of this list. Options always appear in this list in the same order as in the ZOIA option list for the module being added or changed.
40	([Module size]*4) - 56	Additional 0ptions	List of unsigned long integers – After the [Module Option] field there could be any number of additional values stored here. This list can also be empty (see note 1). This seems to depend on the module and what extra values the module needs to store. Additional R&D is required to interpret this completely. So far, they seem to be the value assigned to the module inputs, like the gain for a «Audio Output» module if the gain option is selected.
End - 16	16	Module Name	Text – the module's name (see note 1). This field is optional. On older patches, modules did not have names. This will be zeroes if the field is present and the user did not change the default module name.

Note 1:

Designing a simple way to figure out the number of values in the [Additional Options] field is a bit of a challenge. The reason being that there is more than one field that can affect the [Module Size]. First, there is [Additional Options] which can contain an unknown number of 32 bits values and [Module Name] which can be present or not. Because of that, it cannot be computed directly from the [Module Size] value.

If a [Modules Colors] section if present in the patch file, it means it has been saved with firmware 1.10+ and since the module names came into existence with firmware version 1.04, we can be sure that the [Module Name] is present. In this situation, the number of bytes for the [Additional Options] field can be calculated directly from the [Module Size].

On the other hand, is we cannot be sure whether there is a [Module Name], we propose this method:

1. Compute the number of bytes available from position 40 (start of [Additional Options]) up to the end of the module's data by subtracting 40 from [Module Size].
2. If the number of available bytes is 16 or more, analyze the last 16 bytes to check if it contains only values that are valid for a text. If so, suppose that the [Module Name] field is present. If it contains any invalid value, we know there are no name present.

3. Depending on the outcome of step 2, compute the number of bytes left for the [Additional Options] field.

Connections section

This section provides information on each connection between one module input and output.

Field Offset	Field Length	Field Name	Description
0	4	Number of connections	Unsigned long integer – the number of [Connection definition] that follows.
4	variable	List of [Connection definition]	A list of [Module definition]. The number of modules is given by the previous field. The format of an individual module in this list is described in the table below.

Connection definition

The following table describes the format of each connection. The field offset is relative to the start of the connection definition in the list.

Field Offset	Field Length	Field Name	Description
0	4	Source Module Number	Unsigned long integer – the module number at the origin (source) of the connection. It's an index in the list of modules defined in the [Modules] section.
4	4	Source Block Number	Unsigned long integer – the block index from which the connection starts in the source module. It's an output block (**see note 1**).
8	4	Destination Module Number	Unsigned long integer – the module number where the connection ends. It's an index in the list of modules defined in the [Modules] section.
12	4	Destination Block Number	Unsigned long integer – the block index to which the connection ends in the destination module. It's an input block (**see note 1**).
16	4	Connection Strength	Unsigned long integer. The values are in the range 0 to 10000 (base 10) or 0x0 to 0x0270. On the ZOIA, connection strength can be set either as a dB value or a % value. In dB, the range is –100dB to +12.00dB. In %, the range is 0.001% to 398.1%. The formula to convert the connection strength value to a dB value is: 0 - ((10000 - «connection strength») / 100). The formula to convert the dB value to a % is: 100 * (10 ^ («dB value» / 20)).

Note 1

About [Source Block Number] and [Destination Block Number]: These values are indexes (0 relative) of blocks in the origin and destination modules referenced by the connections. It is **important** to note

that it does NOT refer to the module instance's visible blocks but instead, it refers to the block index in the list of all the blocks provided by the module type, like if all the possible module's blocks were visible. Even though some blocks might be hidden because of the options selected for a specific module.

Pages Names section

This section contains the names assigned to page in the patch, if any.

Note

This section may not contain any page name (the [Number of pages] may be zero) if none of the pages were given a name. This means that we have to look at the [Page Number] field in the modules list to figure out the actual number of pages used in the patch.

Field Offset	Field Length	Field Name	Description
0	4	Number of page names	Unsigned long integer – the number of page names that follows. Note that the value might be zero.
4+(n*16)	16	Page Name	Text – for each page, the page name.

Starred Elements section

Field Offset	Field Length	Field Name	Description
0	4	Number of starred elements	Unsigned long integer – the number of [Starred element definition] that follows. Note that the value might be zero.
4	variable	List of [Starred Element Definition]	A list of [Starred element definition]. The format is described in the table below. Each starred element is a 32 unsigned integer value.

Starred Element definition

The following table describes the format of one starred element in the list field of the [Starred Elements] section.

Starred elements can be applied to either: a) an individual module parameter or b) a connection. We call them parameter-type or connection-type.

A starred element in the list is always described with a 32 bits unsigned integer value but the way it's decoded varies depending of whether it’s a parameter-type of a connection-type starred element.

The most significant bit (bit 31) is **zero** for a parameter-type element and **one** for a connection-type element.

For a parameter-type element, the remaining bits (0-30) are decoded as follows:

- Bits 0-15 => The module index number.
- Bits 16-22 => The input block number in the module (**see note 1**).
- Bits 23-30 => The MIDI CC value (**see note 2**)

For a connection-type element, all the bits in the 32 bits value must be inverted (0 becomes 1 and 1 becomes 0) before the content can be decoded. Once inverted, the bits (0-31) are decoded as follows:

- Bits 0-15 => The connection index number **(see note 3)**.
- Bits 16-22 => not used.
- Bits 23-30 => The MIDI CC value (**see note 2**)

Note 1

The block number follow the same numbering scheme as in the [Connection] section. Refer to that section for more details.

Note 2

The MIDI CC value in bits 23-30 is zero if the starred parameter has no MIDI CC value defined. If the value in bits 23-30 is not zero, the MIDI CC number is obtained by subtracting 1 from that value. In other words, MIDI CC #0 appears as value 1, MIDI CC #1 appears as value 2, etc.

Note 3

The connection index number is a value starting at zero that identify a specific connection in the list of connection from the [Connections] section. Connection zero is the first one, connection one is the second one, etc.

Modules Colors section

This section is optional. To know if it's present you have to compute the combined size of all the previous sections and compare it to the [Patch Size] from the [Patch Header] section to know if there is room left in the patch data after the [Starred Elements] section.

The format for this section is different from the previous ones because there is no field telling us the number of colors present in the section. Instead, it is assumed that the number of colors is the same as the number of modules in the [Modules] section.

Field Offset	Field Length	Field Name	Description
0	4	Color 0	Unsigned long integer – (extended) color for module 0. See appendix B for more details.
4	4	Color 1	Unsigned long integer – color for module 1.
8	4	Color 2	Unsigned long integer – color for module 2.
…			
N*4	4	Color N	Unsigned long integer – color for module N.

Appendix A – Module types

The following table shows the list of [Module type] with their numeric value and their corresponding name and category.

Analysis of a patch definition requires the knowledge of the number and the nature of each block for a particular module instance. This can vary depending on which options are set for a module instance. This appendix will have to be expanded (greatly) in order to provide such information.

Type ID	Module Type	Module Category
0	SV Filter	Audio
1	Audio Input	Interface
2	Audio Output	Interface
3	Aliaser	Audio
4	Sequencer	Control
5	LFO	Control
6	ADSR	Control
7	VCA	Audio
8	Audio Multiply	Audio
9	Bit Crusher	Audio
10	Sample & Hold	Control
11	OD & Distortion	Effect
12	Env Follower	Analysis
13	Delay Line	Audio
14	Oscillator	Audio
15	Pushbutton	Interface
16	Keyboard	Interface
17	CV Invert	Control
18	Steps	Control
19	Slew Limiter	Control
20	MIDI Notes in	Interface
21	MIDI CC in	Interface
22	Multiplier	Control
23	Compressor	Effect

24	Multi-Filter	Audio
25	Plate Reverb	Effect
26	Buffer Delay	Audio
27	All-Pass Filter	Audio
28	Quantizer	Control
29	Phaser	Effect
30	Looper	Audio
31	In Switch	Control
32	Out Switch	Control
33	Audio In Switch	Audio
34	Audio Out Switch	Audio
35	Midi Pressure	Interface
36	Onset Detector	Analysis
37	Rhythm	Control
38	Noise	Audio
39	Random	Control
40	Gate	Effect
41	Tremolo	Effect
42	Tone Control	Effect
43	Delay w/Mod	Effect
44	Stompswitch	Interface
45	Value	Control
46	CV Delay	Control
47	CV Loop	Control
48	CV Filter	Control
49	Clock Divider	Control
50	Comparator	Control
51	CV Rectify	Control
52	Trigger	Control
53	Stereo Spread	Audio
54	Cport Exp/CV in	Interface
55	Cport CV out	Interface

56	UI Button	Interface
57	Audio Panner	Audio
58	Pitch Detector	Analysis
59	Pitch Shifter	Audio
60	Midi Note Out	Interface
61	Midi CC Out	Interface
62	Midi PC Out	Interface
63	Bit Modulator	Audio
64	Audio Balance	Audio
65	Inverter	Audio
66	Fuzz	Effect
67	Ghostverb	Effect
68	Cabinet Sim	Effect
69	Flanger	Effect
70	Chorus	Effect
71	Vibrato	Effect
72	Env Filter	Effect
73	Ring Modulator	Effect
74	Hall Reverb	Effect
75	Ping Pong Delay	Effect
76	Audio Mixer	Audio
77	CV Flip Flop	Control
78	Diffuser	Audio
79	Reverb Lite	Effect
80	Room Reverb	Effect
81	Pixel	Interface
82	Midi Clock In	Interface
83	Granular	Audio
84	Midi Clock Out	Interface
85	Tap to CV	Control
86	MIDI Pitch Bend In	Interface
103	Device Control	Interface

104	CV Mixer	Control

Appendix B – Colors

The following table shows the numeric values for the color parameters used in ZOIA patches.
Since firmware 1.10, a new (extended) set of colors were made available (see the [Modules colors] section).

The table shows both sets of colors since they share the same numeric values. Old colors have numeric values from 0 to 7.

Note that with firmware 1.10 and beyond, when a extended color is assigned to a patch module, the ZOIA also sets the [Old Color] field for the module to the corresponding old color as described in the following table.

Color Number	Color	Corresponding Old Color
0	? Unknown Meaning ?	
1	Blue (*1)	[1] Blue
2	Green (*1)	[2] Green
3	Red (*1)	[3] Red
4	Yellow (*1)	[4] Yellow
5	Aqua (*1)	[5] Aqua
6	Magenta (*1)	[6] Magenta
7	White (*1)	[7] White
8	Orange (*2)	[3] Red
9	Lime (*2)	[2] Green
10	Surf (*2)	[5] Aqua
11	Sky (*2)	[1] Blue
12	Purple (*2)	[6] Magenta
13	Pink (*2)	[3] Red
14	Peach (*2)	[3] Red
15	Mango (*2)	[4] Yellow

*1: this color belongs to the old color set and to the extended color set.
*2: this color belongs to the extended color set only.

Appendix H – Revision History

2021-12 – (marcuslupinus on github) Reformatted the document, corrected some typos, added some context at the beginning and pursued reverse engineering of the patch file format using v 2.50 of ZOIA Firmware. This version includes reverse engineering of these topics: relation between extended and old color, module block numbers, starred parameters interpretation, connections strength interpretation, module options interpretation, module version interpretation and description of the data structure required to decode the module types.

2022-02 - (marcuslupinus on github) Added the fact that ZOIA 'stars' can be applied to module parameters or connections. Prior to this, we thought only individual module parameters could be starred. The decoding of the values of the [Starred Elements] section of the patch file has been adjusted accordingly. Checked correctness with version 2.70 of the firmware.