ruSolut
Data Investigation & Recovery Technology

VISUAL NAND RECONSTRUCTOR

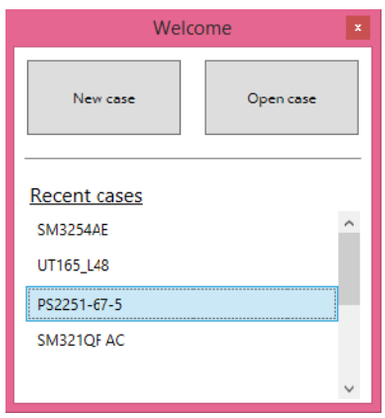
The book

Part2. Software

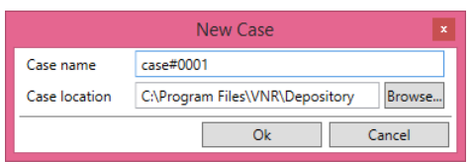
www.rusolut.com

VNR software concept

The Visual Nand Reconstructor software uses a case management system. Each case is stored in a separate folder with physical images. There are three options at software start up - create new case, open case and open recent case.

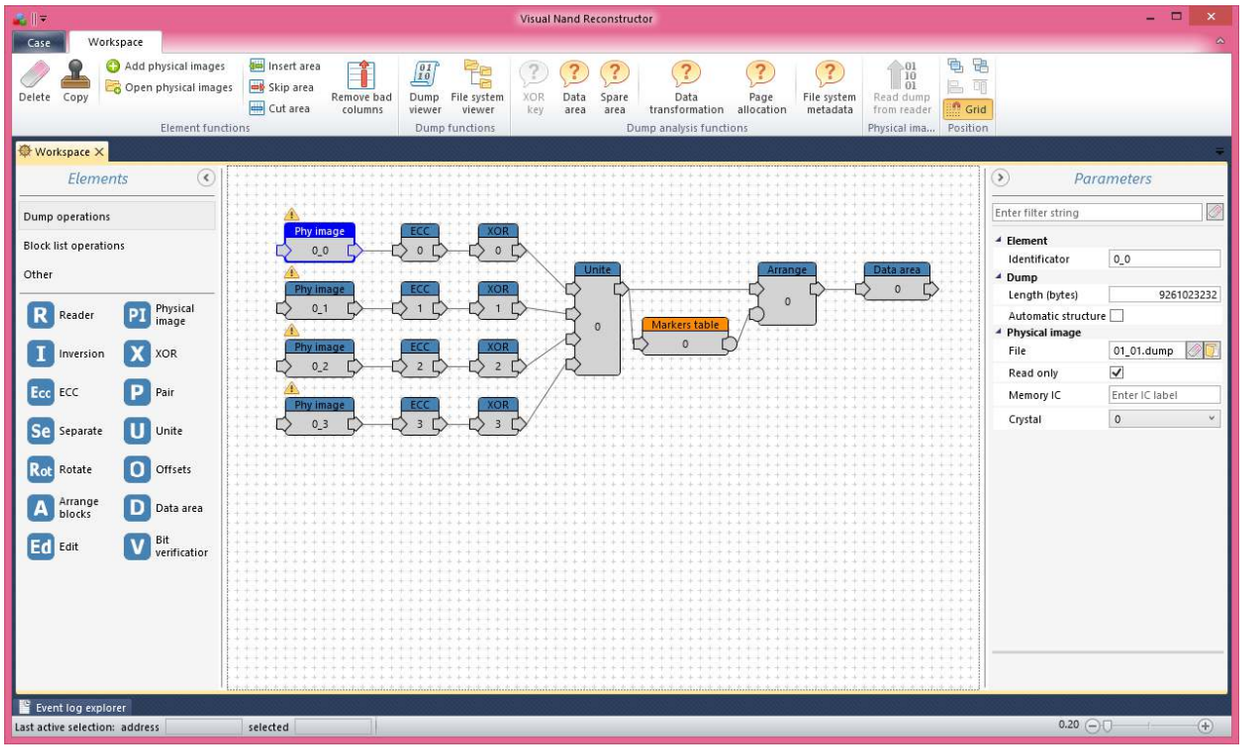


When creating a new case it's necessary to set the path to depository and the case number/name. The subfolder with case name will be created in depository.



The main window of Visual Nand Recontructor is divided into several zones:

Elements (left)
Workspace (center)
Parameters of element (right)
Toolbar (top)
Event log (bottom)



Elements represent the operations for transformation of the physical image of Nand memory chips. Elements are added to the workspace by Drag&Drop method. The set of elements is divided into several types: Dump operations, Block list operations, other elements.

Dump operations contain Reader and Physical image elements, also operations for physical image transformation: ECC, Inversion, XOR, Pair, Separate, Unite, Offsets, Arrange blocks, Data area, etc. Each element has a definite function, which allows transformation of the physical image in accordance with the controller's configuration.

Block list operations contain operations for work with the physical/virtual image on the blocks level. Markers table element is designed for image reconstruction from virtual to logical (translator).

Workspace is a work area, where Elements are added and also parameters and connections between them are set. Almost every element, except the Reader, has an input and output. Element connection is organized the way when each element virtually transforms the source element that connected to it's input. Some elements have two or more inputs, it is necessary for unite of virtual images (crystals or memory chips). All elements have one output, however it's possible to share it between several elements, for example for various hypothesis checking within same case. This ideology is close to the electronic circuit simulation, when every element emulates controller's electronic block.
Conversion from physical image to virtual and then to logical takes place in this workspace. When an element is added, it is necessary to connect it with a previous element, otherwise it won't have data source (except the Reader element). Connections between elements can be deleted and recreated, meanwhile other elements and their parameters stay untouched. To create a connection between elements it's necessary to click on output of source, then to input of the connecting element.

Parameters of element can be set to determine how the image will be transformed. Some of them depend on the number of chips and their physical parameters (page size, block size), some on controller's model (XOR key, ECC). Parameter set is adaptive and depends on the element. All parameters are divided into groups. All parameters are measured in bytes and can be entered in decimal or hexadecimal format and saved automatically while you fullfill appropriate fields.

Toolbar includes automatic and manual physical image analysis modes and other functions. This tab is adaptive. Mode availability depends on the active element. Dump viewer tab contains a number of special modes for image browse, such as: Hex Viewer, Bitmap Viewer, Structure Viewer, Record Viewer. Modes can be used simultaneously and synchronously for unlimited number of physical images.

Events and errors which appear in work process are displayed in Event Log.

Elements

Elements are divided into Dump Operations and Block operations. The Reader and Physical image elements are containers of physical image (direct NAND access and dump file). All other Dump Operation elements virtually modify physical image image in accordance with parameters. Markers Table element sets parameters of translation of physical/virtual blocks into logical image.

The Reader element is a program module of NAND Reader. It is designed for realtime NAND memory access and physical image extraction. When the new case created the Reader is added automatically.

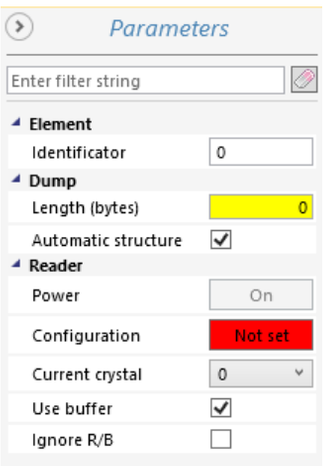
The set of functions for reader is available on the adaptive Toolbar.

Read ID allows to get identifier of NAND chip.

Read ONFI chip configuration allows to read memory chip's parameters from special page, in case if NAND chip conforms to ONFI specification. This works well for Micron (0x2C) and Intel (0x89) chips.

Read bad columns allows to read defective columns (bytes) which were programmed in NAND chip at factory. This option is currently supported for some Sandisk and Toshiba NAND chips.

The Parameters tab contains settings of NAND chip access.



NAND chip's access parameters must be set in Configuration from built-in database

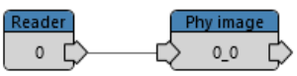
Current crystal represents the active crystal (CE) of NAND from which the dump will be read/accessed.

Power ON/OFF turns the power of NAND chip ON/OFF. Pressing the button performs the action displayed on it (not status!). When power is ON, the yellow LED of reader is alight.

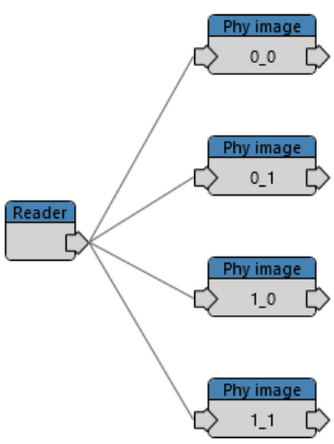
Ignore R/B function is used when analyzing new NAND chip configurations. It helps to avoid reader's hang up. However, it's not recommended to use it normally.

Detailed instruction how to use Reader element to access NAND and extract physical images can be found in the web article:
http://rusolut.com/direct-access-to-nand-and-physical-image-extraction/

The Physical Image element contains a binary copy of the NAND chip. When extracting physical image (dump reading), the RAW NAND data is recorded into a dump file on the disk.



Every crystal of NAND chip is represented by a Physical Image element when extracting dump. For example, in case of two NAND chips and two crystals per each, it is necessary to add 4 Physical Image elements.

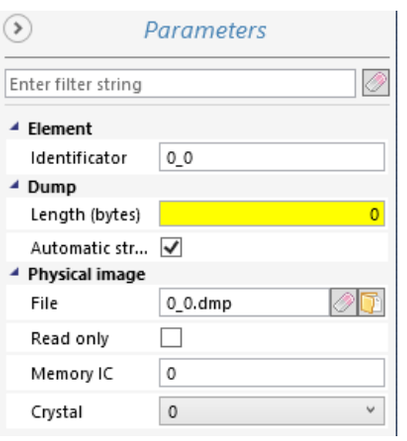


Physical Image element has input and output. The input is used for reader connection. The data goes out of reader to the Physical Image element and saved to dump file. On the output Physical Image connects with other elements for further transformation of physical image to virtual and logical.

The set of functions for Physical image element is available on the adaptive Toolbar.

Read dump operation starts physical image extraction process.

The Parameters tab contains settings of Physical image element.



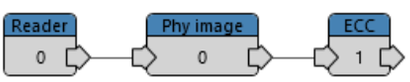
File contains the path to file with physical image (dump file).

Read only is set to protect dump files from modification through hex viewer.

Crystal represents the crystal (CE) of NAND this pysical image belongs to.

The ECC element (Error Correction Code) allows to correct bit errors which appear during data recording/reading process in flash memory and also while physical image extraction. Uncorrected bit errors damage the user's data and corrupt files. The ECC decoder corrects errors on-fly, using code stored in pages of NAND chip.

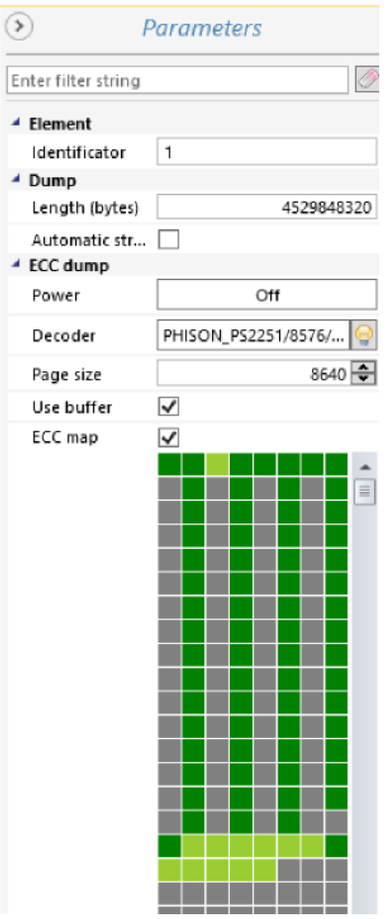
ECC element must be connected to the Physical Image, or after elimination of Bad Columns if they exist.



The set of functions for ECC element is available on the adaptive

Reread dump allows to re-read pages of the NAND chip, which can not be corrected due to too high level of errors. If the chip was read with the standard voltage 3.3V and the capability of ECC code is not enough for data correction, it's possible to reduce the voltage to 2.5V ...1.8V in order to reduce level of internal noise of chip. Having lowered the noise, this function does read only pages that were not corrected at previous read attempt. Multiple reread iterations are supported, if image was not ideally reread after first attempt. To use this option it's necessary to remove read only flag from physical image element. Reader must be connected to the Physical image elements.

The Parameters tab contains settings of ECC element.



Power ON/OFF turns ON/OFF bit error correction. Pressing the button performs the action displayed on it (not status!).

Decoder defines ECC code format used for particular controller. Different controllers use different code types. Decoder can be selected manually or automatically, by pressing on button 

Page size must be adjusted to the NAND's page size.

ECC map allows to estimate the quality of data correction. One square represents one page.

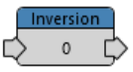
Dark green = good page (no errors and no correction required)

Light green = corrected page (all errors corrected)

Red = bad page (too much errors, correction doesn't work due to code's power limit)

Grey = empty page (filled with FFFF)

Note: pressing on ECC map flag does not enable correction! In order to enable dump correction turn power ON.

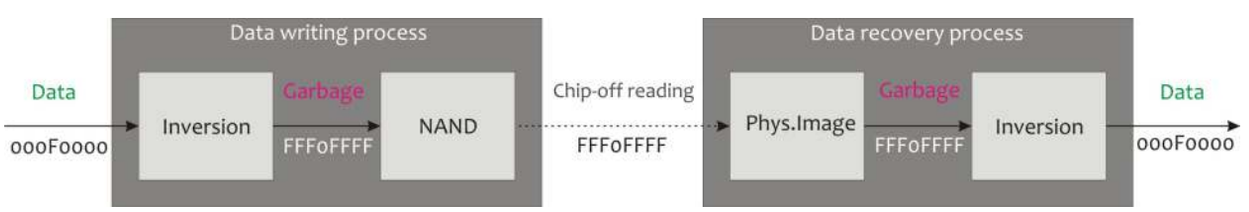


Inversion element performs the binary operation – NOT. Inversion converts the physical image in accordance with a simple binary rule:

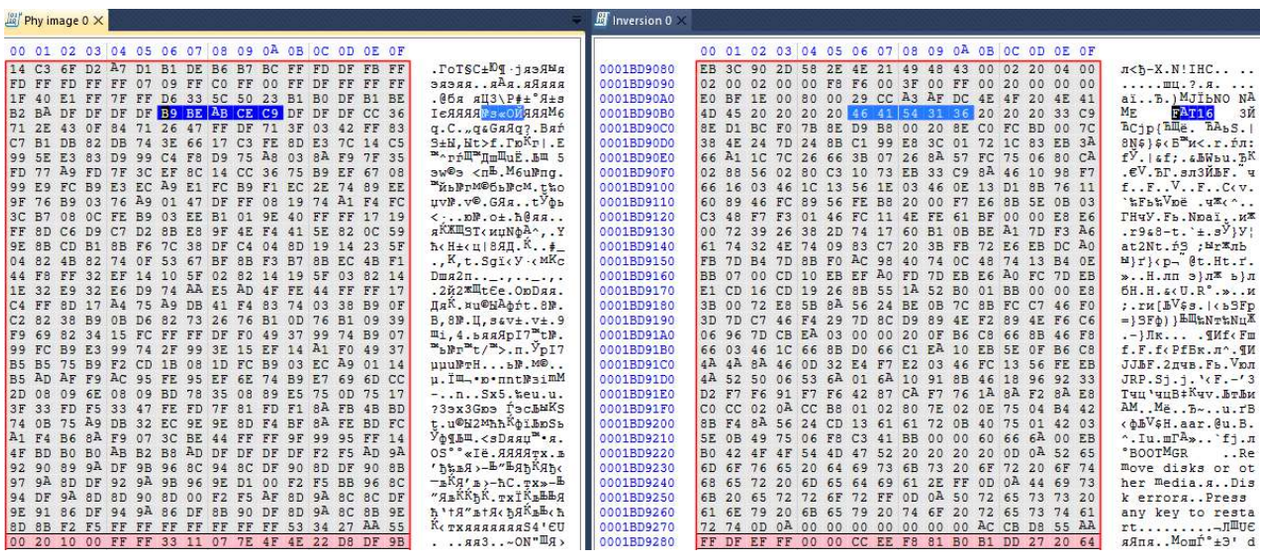
Not 0 = 1
Not 1 = 0

Some controllers invert data before recording into flash memory, to minimize the wear of memory cells. To convert inverted physical image to normal state, it's necessary to apply inversion.

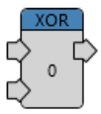
The functional scheme of Inversion in working flash storage device and reverse process is represented below



In hexadecimal format it looks like on the picture below. The inverted data is on the left, the original data is on the right.



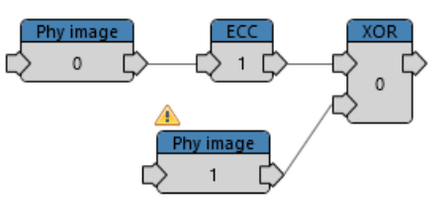
The Inversion element has no adjustable parameters.



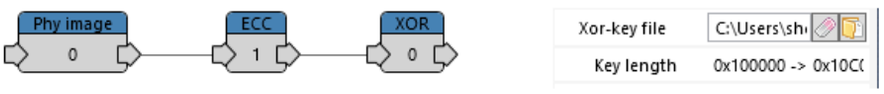
The XOR element decrypts the data that was scrambled (XOR'd) in controller's data transfer channel. This mathematical operation works according to the rule:

Data	XOR key	XOR'd data
0	0	0
0	1	1
1	0	1
1	1	0

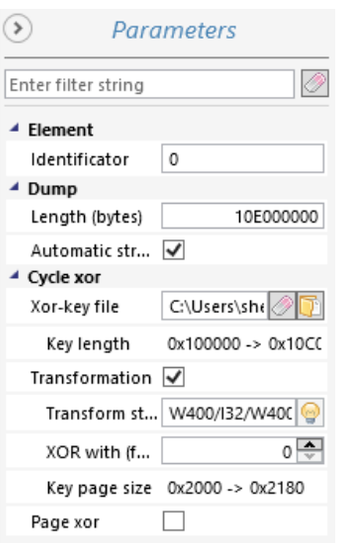
The XOR element has 2 inputs and one output. The physical/virtual image is connected to upper input, the physical image element with XOR key loaded as file is connected to lower input. This method of connection of the XOR element is used for custom built XOR keys and new XOR key analysis and extraction.



When a XOR key is supported and available in VNR resources, it must be selected from Parameters tab of XOR element. Then the second lower input will automatically disappear.



The Parameters tab contains settings of XOR element.



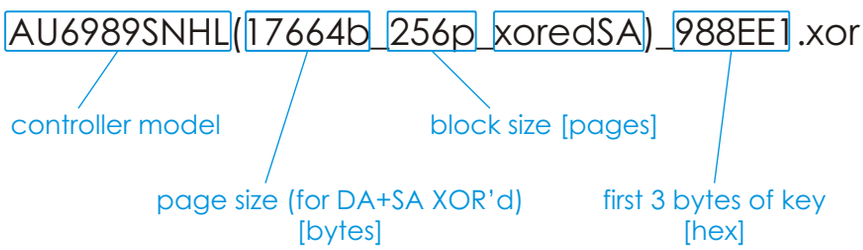
XOR-Key file contains the path to XOR key file.

Transformation flag is used for XOR key adjustment, according to page structure. There are 2 different XOR key formats - XOR Key for Data area only, and XOR Key for Data area and Spare area. Transformation flag is required for XOR Keys which applied to Data area only (this type is used in ~90% of all controllers). When XOR Key applied to Data area and Spare area, transformation flag must be unmarked.

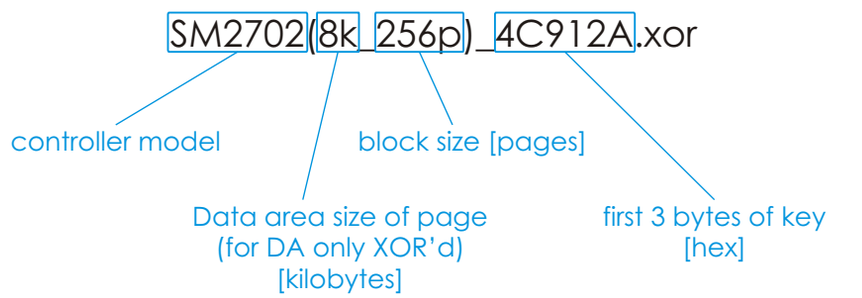
The button automatically adjusts the XOR key structure according the page structure (when XOR key for Data area only is used)

XOR Keys have two formats - for Data area only and for Data area with Spare area. All their parameters which should match to the given case are mentioned in XOR key file names.

XOR key for xor'd Data area and Spare area

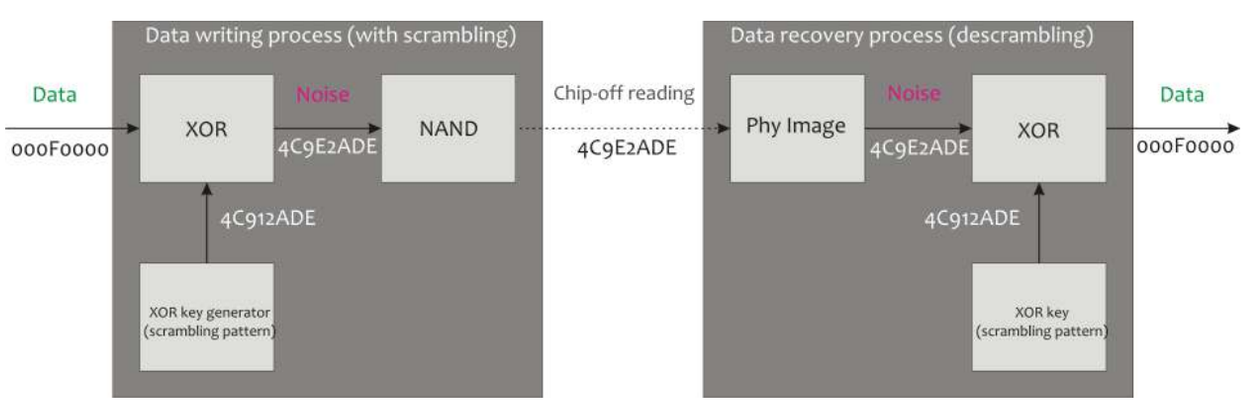


XOR key for xor'd Data area only

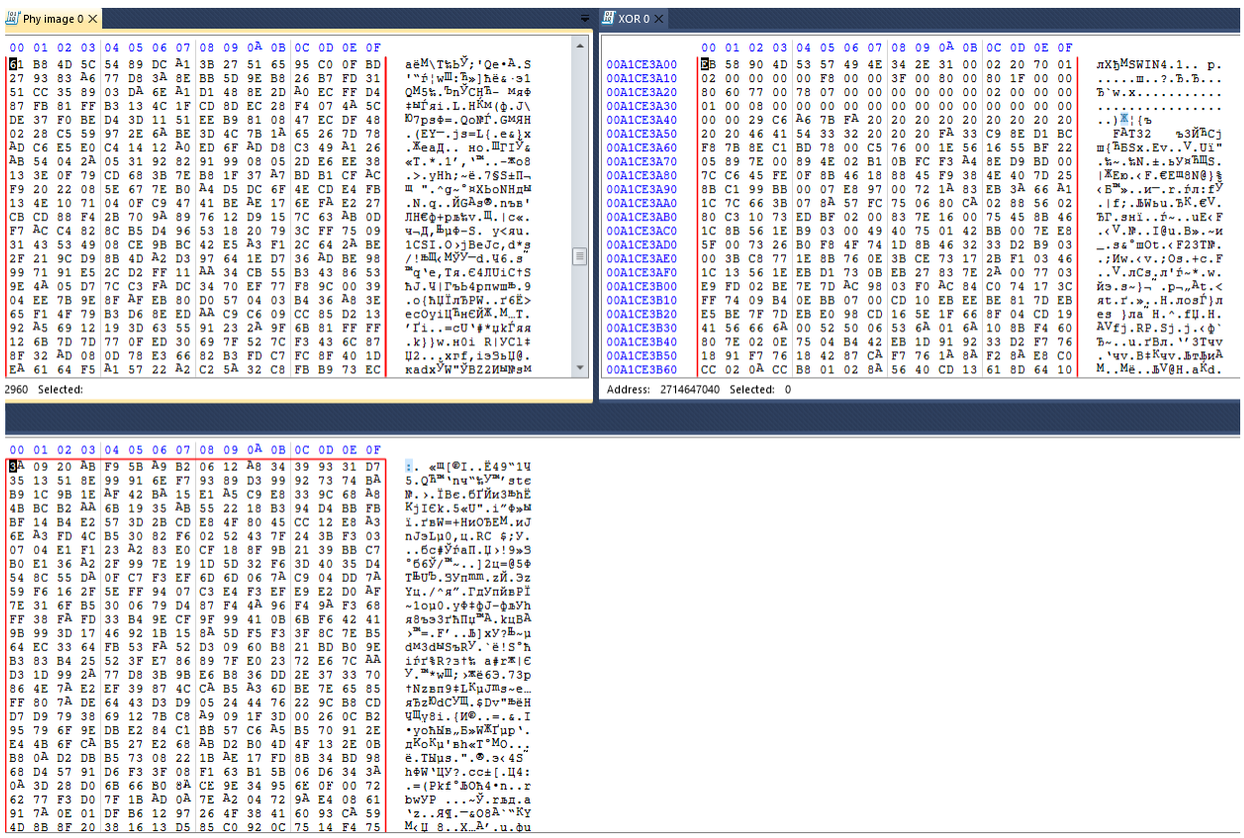


Every controller model may use one of 2-3 keys, depending on the page and block size of NAND chip

The functional scheme of XOR in working flash storage device and reverse process is represented below



In hexadecimal format it looks like on picture below. The XOR'd data is on the left, the decrypted data is on the right, the XOR key is at the



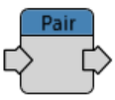
In Bitmap it looks like on following picture. The XOR'd data is on the left, the decrypted data is on the right, the XOR key is at the bottom

XOR'd data looks like the noise and has no patterns.

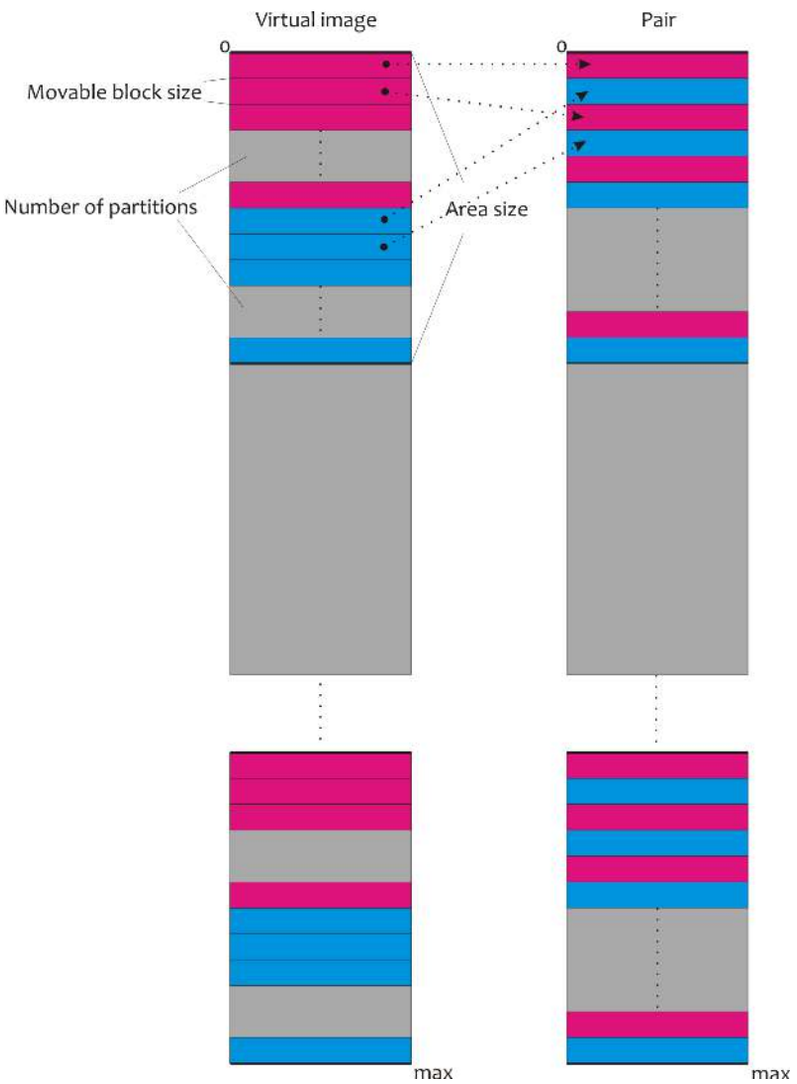
Decrypted or original data usually has different horizontal patterns.

XOR key has very specific patterns, depending on controller it may have totally different look.

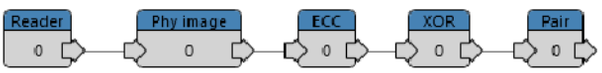
The online library of XOR key patterns can be found on the website:
http://rusolut.com/xor-key-library/



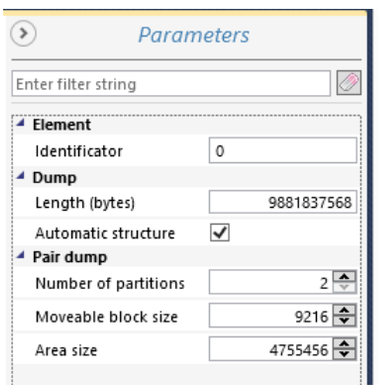
The Pair element performs a page reordering within virtual block, according to multi-plane block allocation scheme.



The Pair element must be connected to the virtual image at the end of scheme, if controller used multi-plane block allocation scheme (used in 90% of cases)



The Parameters tab contains settings of Pair element.

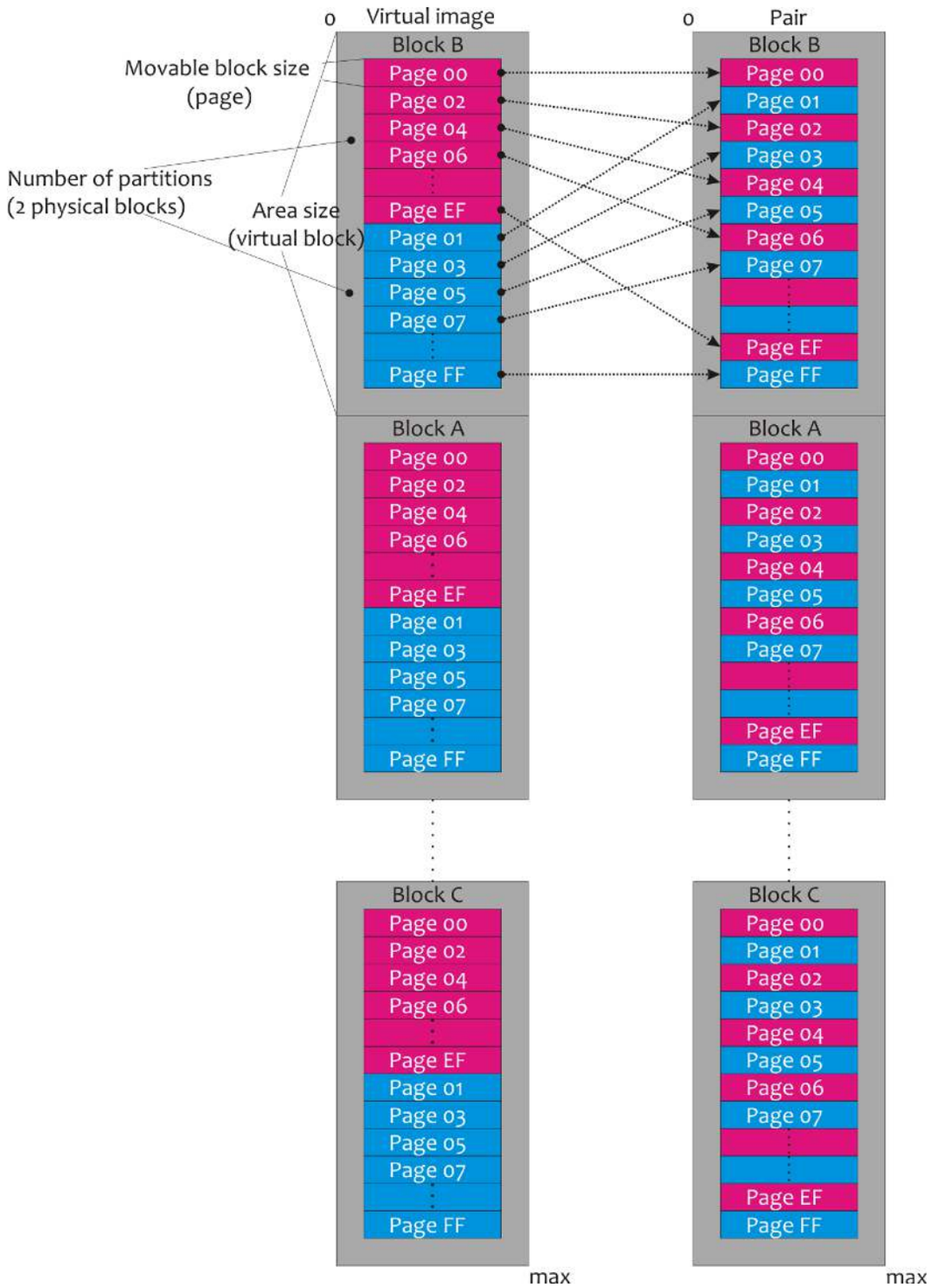


Number of partitions defines the number of parts on which the Area will be divided. Normally it's equal 2, sometimes 4, depending on how many physical blocks used for virtual block allocation (2-plane and 4-plane interleaving)

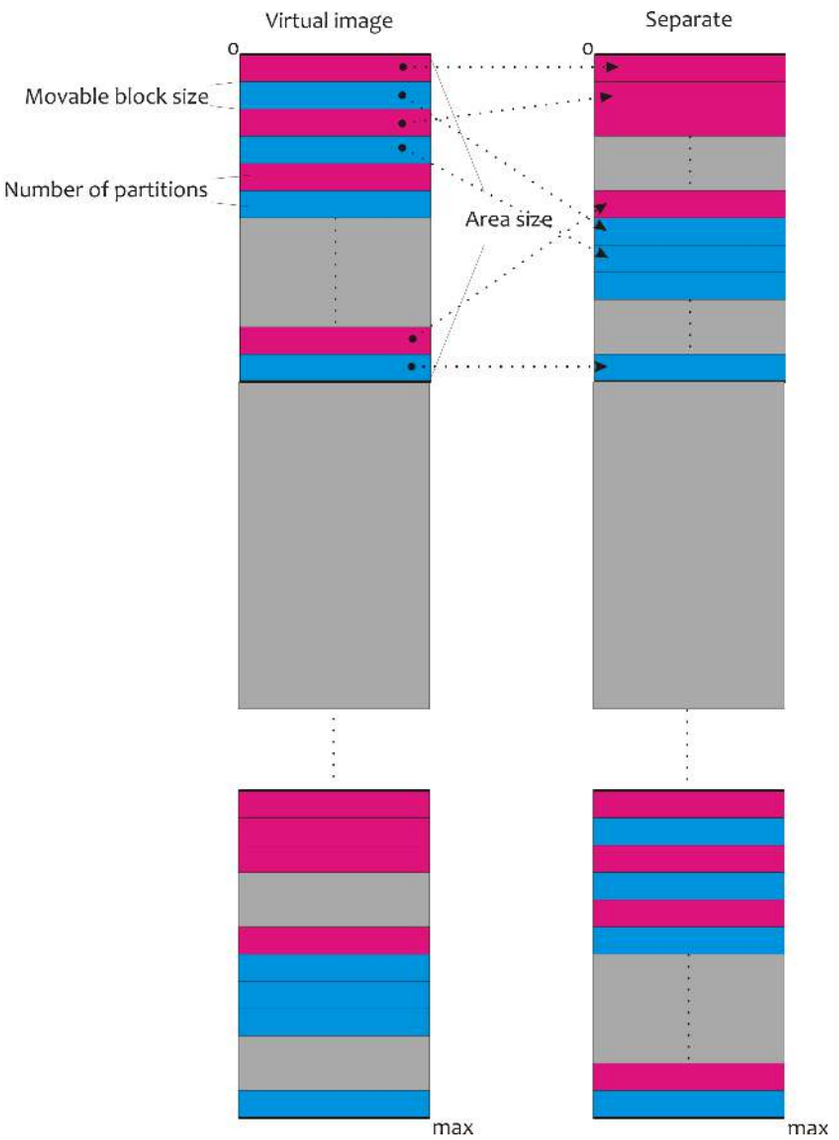
Movable block size determines the area size that is being reordered inside the Area size. Normally it's equal to page size of NAND chip.

Area size defines the periodic area, inside of which there are reorderings of movable areas. Normally it's equal to virtual block size (virtual block consists of 2 or 4 physical blocks, depending on multi-plane block allocation scheme).

The Pair operation for 2-plane block allocation scheme (virtual block = 2 physical blocks) is shown below:

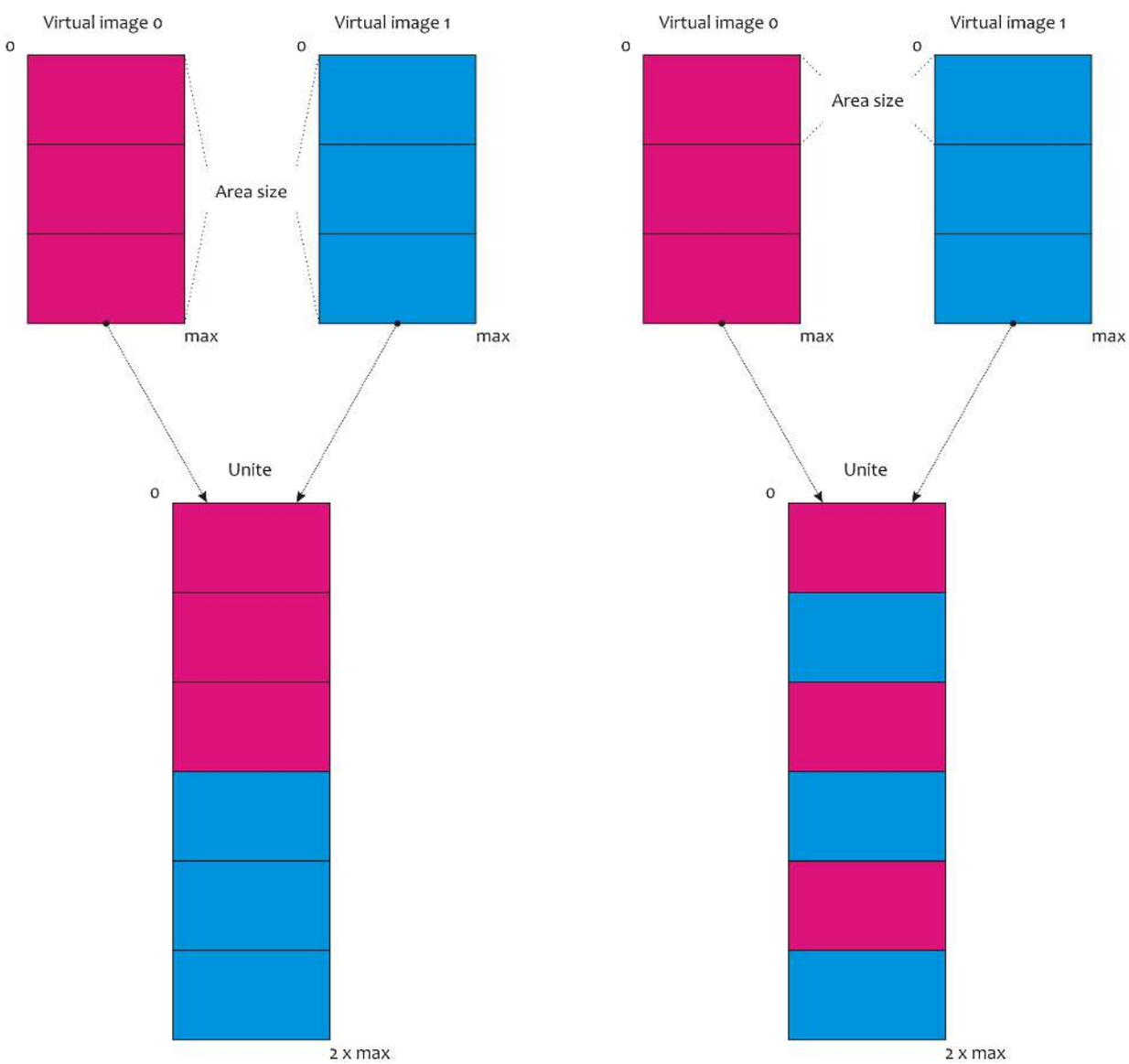


The Separate element performs a page reordering within virtual block, reverse to Pair element.

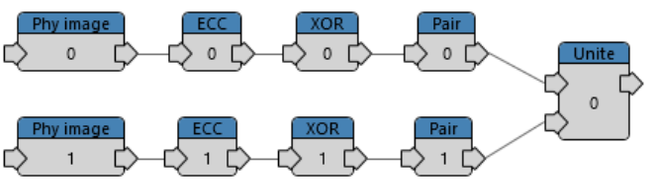


It has same parameters as Pair element

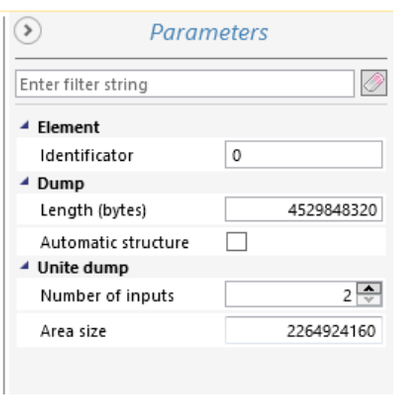
The Unite element joins virtual images together (dumps of crystals, NAND chips) with specific step. This operation is used to join NAND chips and crystals of one chip, also during Bad Column removal. When more than one physical image presented in case, they all must be united at the end of analysis to build the logical image, according to block allocation scheme (sequentially or parrallel).



The Unite element joins the virtual images at the end of analysis.

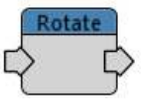


The Parameters tab contains settings of Unite element.

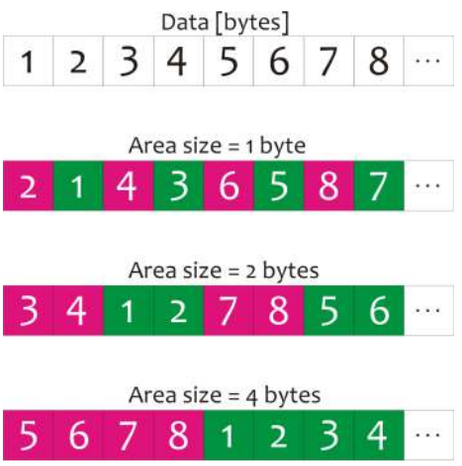


Number of inputs defines the number of dumps which will be united periodically (Area size).

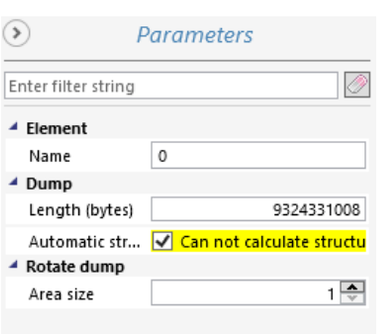
Area size defines the periodic area that is taken from several dumps and joint together. Normally it's equal to page size (parallel) or dump size (sequentially). Sometimes it may be equal to 1 or 8 bytes, depending on controller's block allocation scheme.



The Rotate element changes byte order (group of bytes) in the whole dump.

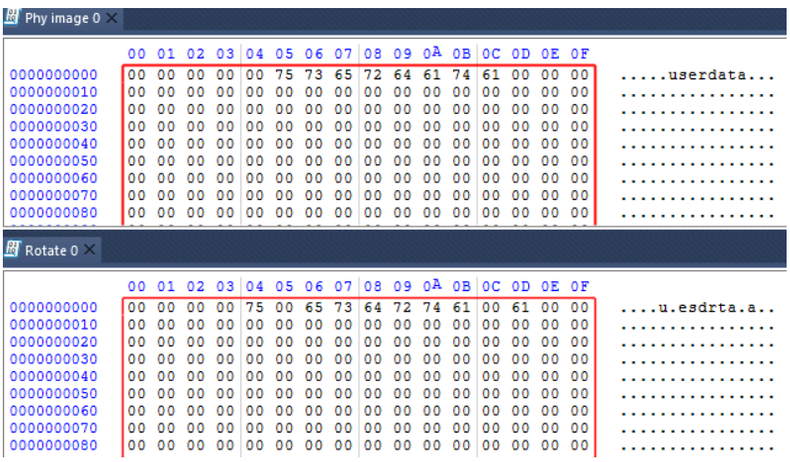


The Parameters tab contains settings of Rotate element.

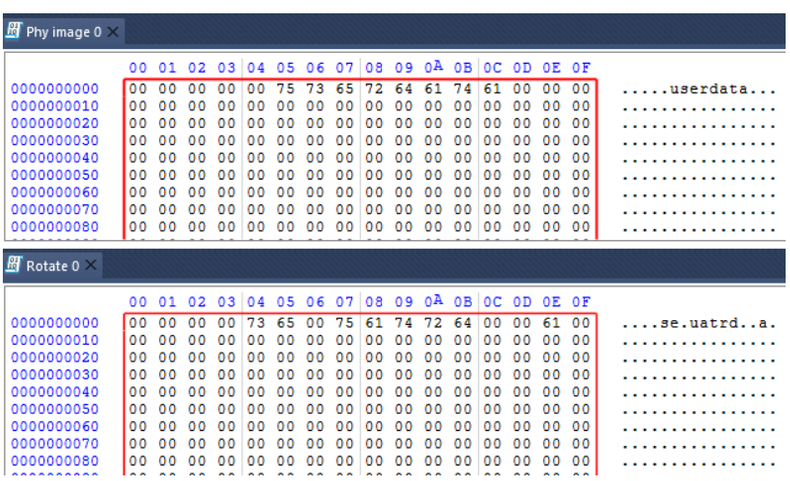


Area size defines the period of rotation, in

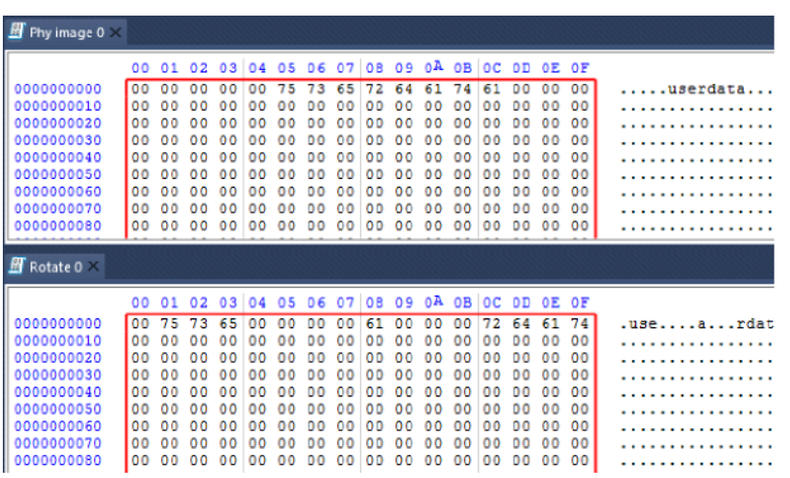
Area size = 1 byte



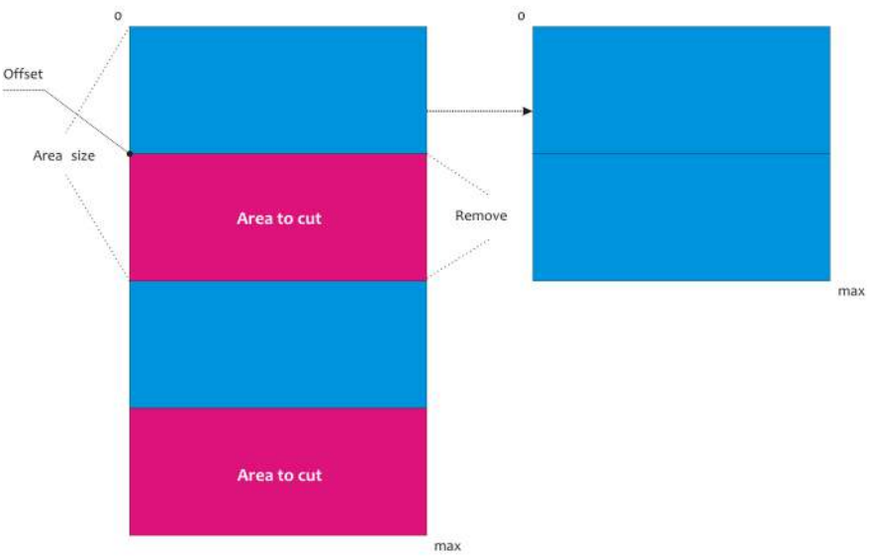
Area size = 2 bytes

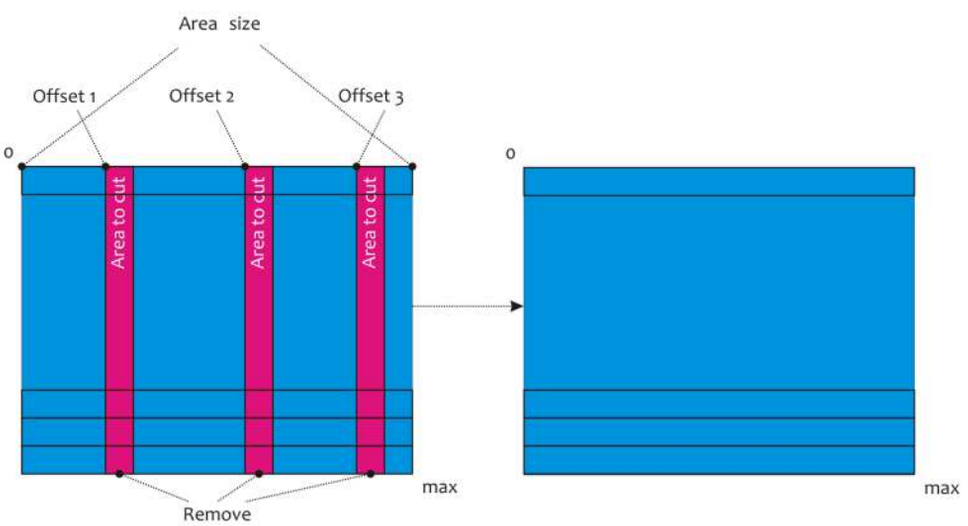


Area size = 4 bytes

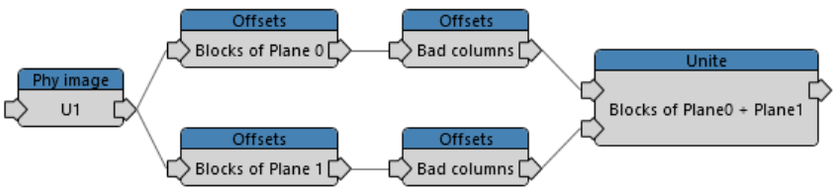


The Offsets element allows to cut/add bytes at specific offsets within periodical areas.

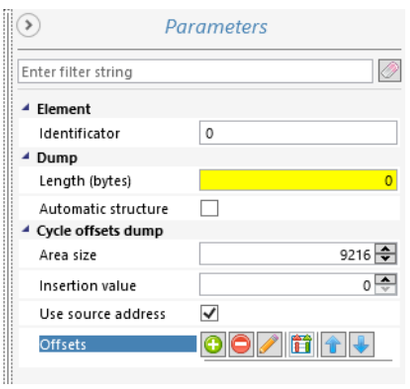




The Offset element used for Bad Column removal operation, as well as for some other non-standard transformations.



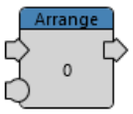
The Parameters tab contains settings of Offsets element.



Area size defines the periodical area where offsets are added. It equals to physical block/page size during bad column removal operation.

Insertion value defines the pattern which will fill the added bytes (this option is not used for data recovery purpose usually).

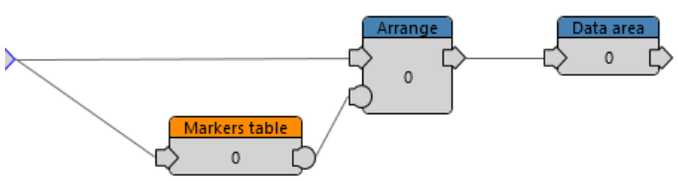
Offsets option is used for manipulations with the physical image - cut/add bytes, edit offset, remove bad column. Number of offsets is unlimited, they can be sonnected in sequence for complex transformations.



The Arrange blocks element is the container of logical image. It is the result of block translation from virtual image to logical through markers table.

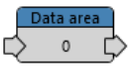
The Arrange blocks element has two inputs and one output. The source element (XOR, Pair, Unite or any other) must be connected to the upper input, the Markers table element to lower input (it sets the blocks order - translator)

The Arrange blocks element must be added in combination with Markers table and Data Area elements.



This element has no adjustable parameters.

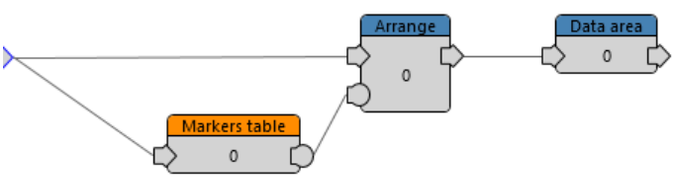
The logical image is shaped inside the Arrange blocks element (including Data Area and Spare Area of pages), after analysis and block table creation via Markers table.



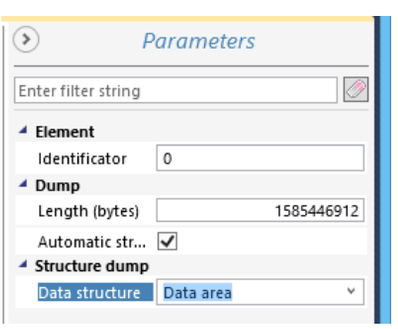
The Data area element is designed to extract the data area of page from logical image, stored in Arrange blocks element.

The Data area element has one input and output, it is the final element in process of physical image transformation. To save the user's data, the File System viewer must be opened on Data area element. It is possible to save the logical image to binary file from this element, for further analysis in forensic tools, using the functions from dump viewer menu (Save all).

The Data area element must be added in combination with Arrange blocks and Markers table elements.



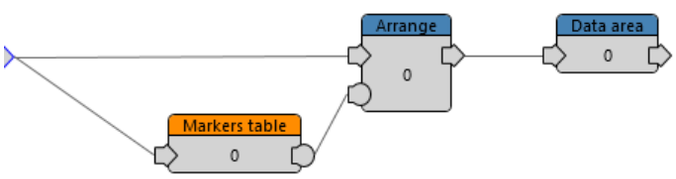
The Parameters tab contains settings of Data area element.



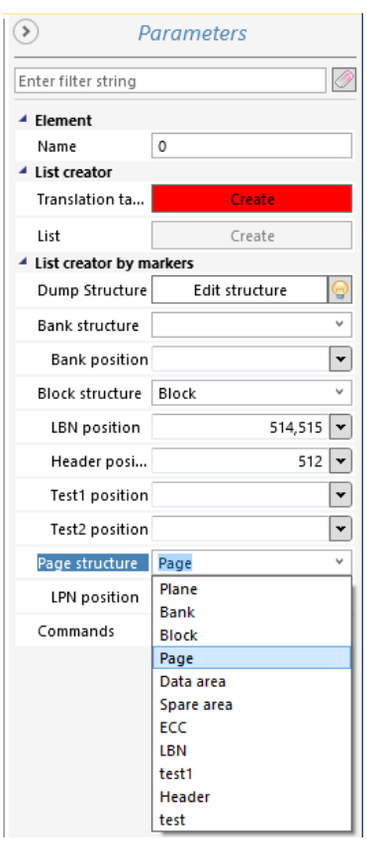
Data structure defines the data area structure of page. it must be predefined in structure viewer and then choosen from this menu.

The Markers table element is designed for physical/virtual block translation into logical image, according to spare area of page and it's parameters (positions of different markers - LBN, Header, etc.). These parameters are set in Structure viewer mode at the step of dump structure analysis.

The Markers table element must be added in combination with Arrange blocks and Data area elements.



The Parameters tab contains settings of the Markers table element.



Dump structure must be copied from source-element using button 

The Block structure is essential and it must be set from drop-down menu.

The Page structure is essential and it must be set from drop-down menu.

The Bank structure is optional, depends on how many banks NAND space is divided. It can be set from drop-down menu.

The LBN structure is essential and it must be set from drop-down menu. In case if LBN has reverse order (e.g.1025,1024), it can be changed manually in the LBN field.

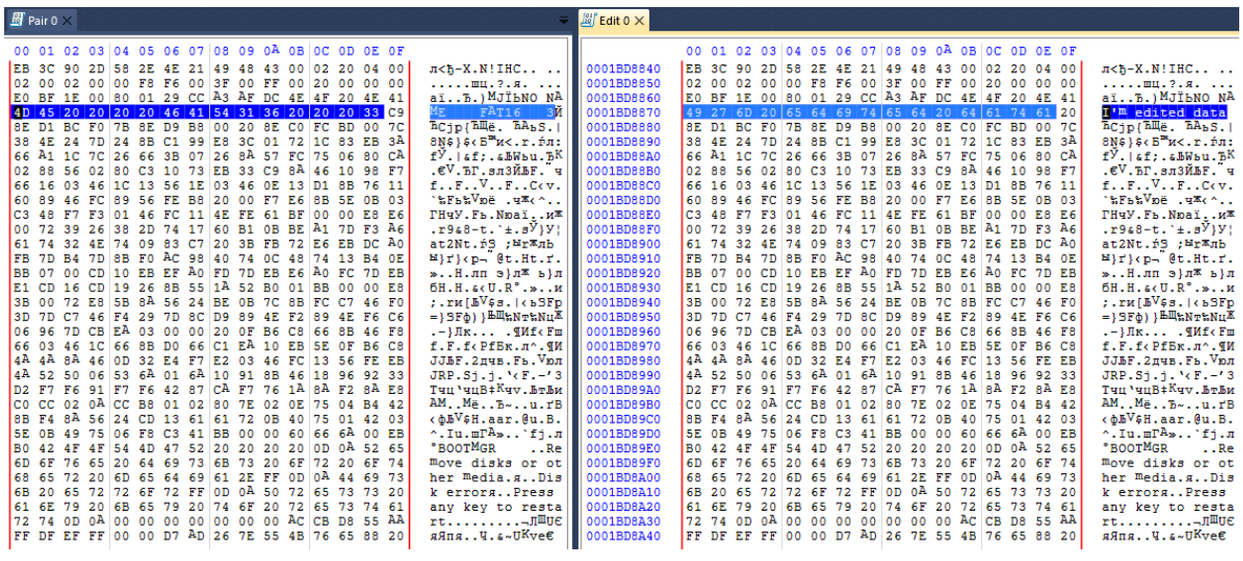
The Header structure is usually essential and it can be set from drop-down menu.

Other structures are optional and can be set from drop-down menu.

The Create translation table button is used to build table of virtual blocks (logical block distribution across physical blocks with mixed order) for further analysis in block table and block re-ordering and filtering.

The Create List button is used to build the logical image in Arrange blocks element (after sorting and filtering blocks according to LBN).

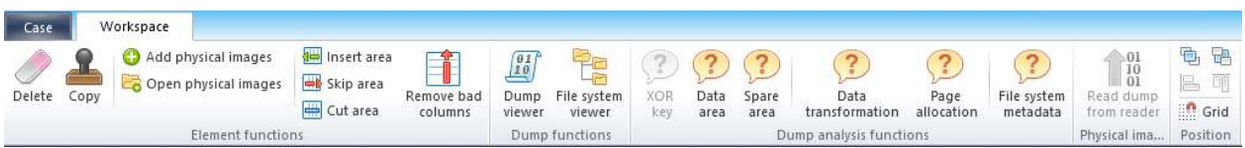
The Edit element virtually changes dump. This operation doesn't bring any changes directly to dump-file of physical image, it does virtual modification. It is useful in cases when some parameters of file system must be changed and other non-standard situations (e.g. to change sector size value in boot sector from 2048b to 512b).



This element has no adjustable parameters.

Toolbar

Toolbar contains a set of tools for work with elements. The set of tools is activated when one of the elements is chosen. Available modes depend on chosen element. All functions are divided by several groups



Element functions (available for all elements)

Reader functions (available for Reader element)

Physical Image functions (available for Physical image element)

ECC functions (available for ECC element)

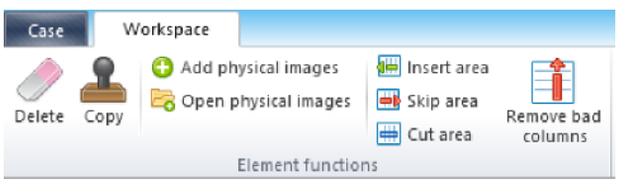
Dump analysis functions (available for elements except Markers table)

Dump functions (available for all elements except Markers table)

Block list operations (available for Markers table element)

Element functions

The group of Element functions contains Delete, Copy, Add physical image, Open physical image, Remove bad columns operations.



Delete operation is used to delete elements and connections between them.

Copy operation is used to clone elements with their parameters. It is used when the same operation applied to multiple dumps.

Add physical images operation is used to add empty physical image elements for further chip reading into it. It is used when the new task is being created.

Open physical images operation is used to import dumps to the task. It is used when the task is created from old dumps that have been previously read.

Remove bad columns is used to remove bad columns automatically, from extracted bad column table of chip (Read bad columns operation)

Reader functions

The group of Reader functions contains Read ID, Read ONFI chip configuration, Read bad columns operations.

All operations available for Reader are described in element description (Elements section)

Physical Image functions

The group of Physical image functions contains Read dump from Reader operation.

All operations available for physical image element are described in element description (Elements section)

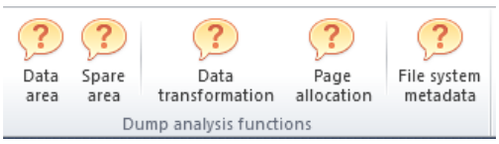
ECC functions

The group of ECC functions contains Reread dump operation.

All operations available for ECC are described in element description (Elements section)

Dump analysis functions

The group of Dump analysis functions contains Data area analysis, Spare area analysis, Data transformation analysis, Page allocation analysis, File system metadata analysis operations.



Data area analysis is used for automatic analysis of page structure. It is used on the stage of physical image structure description.

Spare Area analysis is used for statistical analysis of Spare Area. It is used for detection of LBN, Header and other structures.

Data transformation analysis is used for automatic analysis of Inversion, Unite by byte, Rotation presence or absence. It is used for finding transformation of user's data in the controller-NAND data transfer channel.

Page allocation analysis is used for determination of type of the virtual block allocation, inside the image and also between chips/crystals. It is applied to single dump for analysis of Pair operation (Multi-plane block allocation) and to several selected images (serial/parallel block allocation schemes determination).

File system metadata analysis is used for search and analysis of file system structures (FAT table/MFT File records). It is used for analysis of page allocation inside/between chips. Analysis based on FAT tables/MFT record sequence.

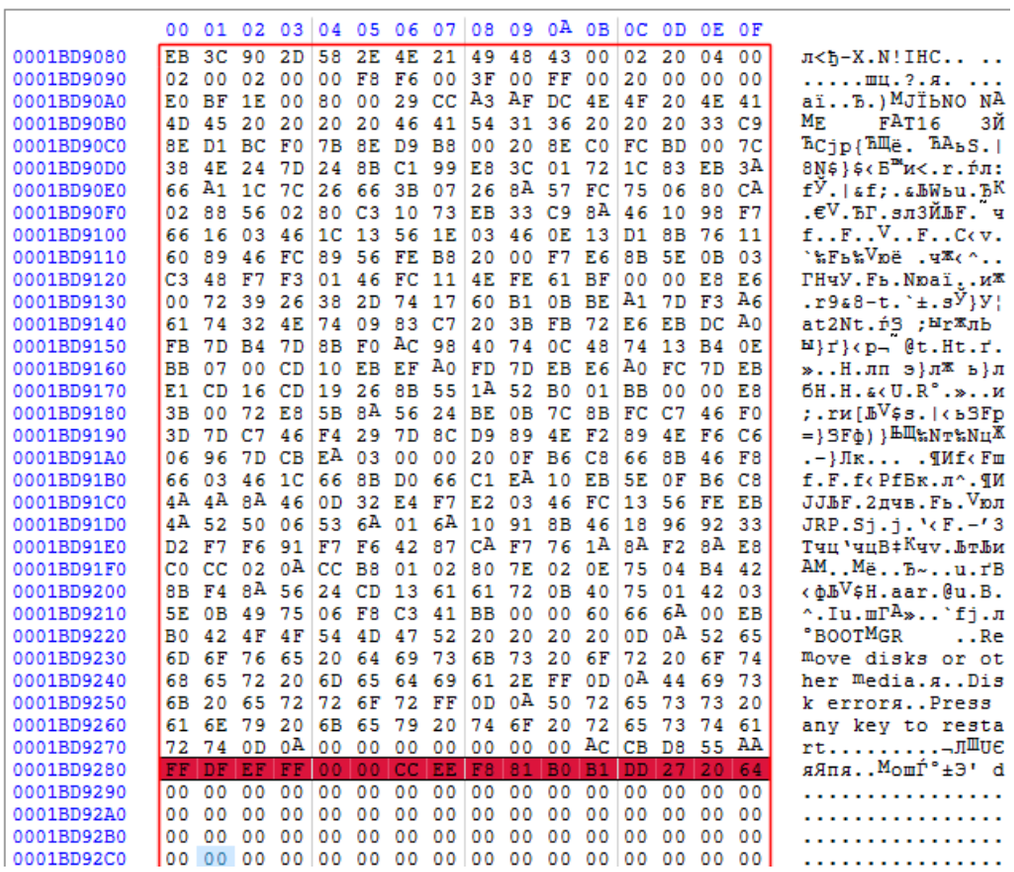
Dump functions

The group of Dump functions includes Dump viewer and File System viewer.

File System viewer is designed for file system browsing and file saving from the logical image.

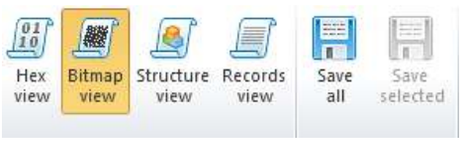
Dump viewer is designed for visual analysis of the content of physical and virtual images in different data formats. It has several data respresentation and view modes.

The Hex View mode is the typical hexadecimal representation of data in physical image.



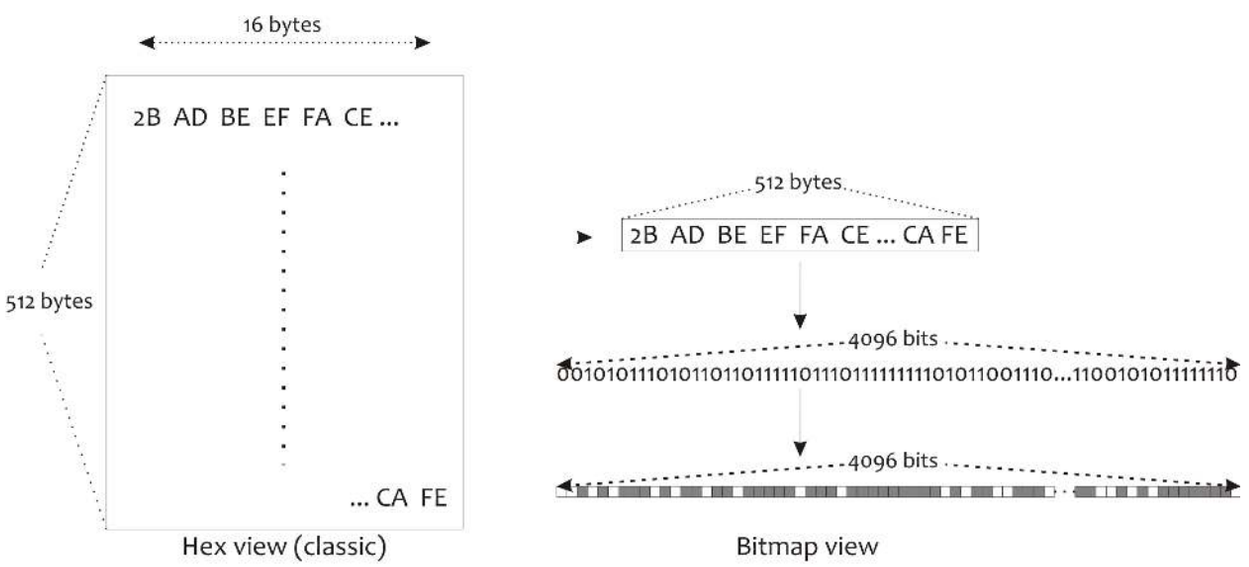
	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	
0001BD9080	EB	3C	90	2D	58	2E	4E	21	49	48	43	00	02	20	04	00	л<ђ-X.N!IHC.. ..
0001BD9090	02	00	02	00	00	F8	F6	00	3F	00	FF	00	20	00	00	00	.....шц.?.я. ...
0001BD90A0	E0	BF	1E	00	80	00	29	CC	A3	AF	DC	4E	4F	20	4E	41	аї..Ъ.)МЈЇЬNO NA
0001BD90B0	4D	45	20	20	20	20	46	41	54	31	36	20	20	20	33	C9	ME FAT16 3Й
0001BD90C0	8E	D1	BC	F0	7B	8E	D9	B8	00	20	8E	C0	FC	BD	00	7C	ЋCjp{ЋЩё. ЋАьS.|
0001BD90D0	38	4E	24	7D	24	8B	C1	99	E8	3C	01	72	1C	83	EB	3A	8N$}$‹Б™и<.r.ѓл:
0001BD90E0	66	A1	1C	7C	26	66	3B	07	26	8A	57	FC	75	06	80	CA	fЎ.|&f;.&ЉWьu.ЂК
0001BD90F0	02	88	56	02	80	C3	10	73	EB	33	C9	8A	46	10	98	F7	.€V.ЂГ.sл3ЙЉF.˜ч
0001BD9100	66	16	03	46	1C	13	56	1E	03	46	0E	13	D1	8B	76	11	f..F..V..F..C‹v.
0001BD9110	60	89	46	FC	89	56	FE	B8	20	00	F7	E6	8B	5E	0B	03	`‰Fь‰Vюё .ч‹^..
0001BD9120	C3	48	F7	F3	01	46	FC	11	4E	FE	61	BF	00	00	E8	E6	ГНчУ.Fь.Nюaї..и‹
0001BD9130	00	72	39	26	38	2D	74	17	60	B1	0B	BE	A1	7D	F3	A6	.r9&8-t.`±.s}У¦
0001BD9140	61	74	32	4E	74	09	83	C7	20	3B	FB	72	E6	EB	DC	A0	at2Nt.ѓЗ ;ыr‹лЬ
0001BD9150	FB	7D	B4	7D	8B	F0	AC	98	40	74	0C	48	74	13	B4	0E	ы}ґ}‹р¬˜@t.Ht.ґ.
0001BD9160	BB	07	00	CD	10	EB	EF	A0	FD	7D	EB	E6	A0	FC	7D	EB	»..Н.лп э}л‹ ь}л
0001BD9170	E1	CD	16	CD	19	26	8B	55	1A	52	B0	01	BB	00	00	E8	бН.Н.&‹U.R°.»..и
0001BD9180	3B	00	72	E8	5B	8A	56	24	BE	0B	7C	8B	FC	C7	46	F0	;.rи[ЉV$s.|‹ьЗFр
0001BD9190	3D	7D	C7	46	F4	29	7D	8C	D9	89	4E	F2	89	4E	F6	C6	=}ЗFф)}ЊЩ‰Nт‰NцЖ
0001BD91A0	06	96	7D	CB	EA	03	00	00	20	0F	B6	C8	66	8B	46	F8	.–}Лк... .¶Иf‹Fш
0001BD91B0	66	03	46	1C	66	8B	D0	66	C1	EA	10	EB	5E	0F	B6	C8	f.F.f‹PfБк.л^.¶И
0001BD91C0	4A	4A	8A	46	0D	32	E4	F7	E2	03	46	FC	13	56	FE	EB	JJЉF.2дчв.Fь.Vюл
0001BD91D0	4A	52	50	06	53	6A	01	6A	10	91	8B	46	18	96	92	33	JRP.Sj.j.‘‹F.–’3
0001BD91E0	D2	F7	F6	91	F7	F6	42	87	CA	F7	76	1A	8A	F2	8A	E8	Тчц‘чцВ‡Кчv.ЊтЉи
0001BD91F0	C0	CC	02	0A	CC	B8	01	02	80	7E	02	0E	75	04	B4	42	АМ..Мё..Ђ~..u.ґВ
0001BD9200	8B	F4	8A	56	24	CD	13	61	61	72	0B	40	75	01	42	03	‹фЉV$Н.aar.@u.B.
0001BD9210	5E	0B	49	75	06	F8	C3	41	BB	00	00	60	66	6A	00	EB	^.Iu.шГA»..`fj.л
0001BD9220	B0	42	4F	4F	54	4D	47	52	20	20	20	20	0D	0A	52	65	°BOOTMGR ..Re
0001BD9230	6D	6F	76	65	20	64	69	73	6B	73	20	6F	72	20	6F	74	move disks or ot
0001BD9240	68	65	72	20	6D	65	64	69	61	2E	FF	0D	0A	44	69	73	her media.я..Dis
0001BD9250	6B	20	65	72	72	6F	72	FF	0D	0A	50	72	65	73	73	20	k errorя..Press
0001BD9260	61	6E	79	20	6B	65	79	20	74	6F	20	72	65	73	74	61	any key to resta
0001BD9270	72	74	0D	0A	00	00	00	00	00	00	00	AC	CB	D8	55	AA	rt.........¬ЛШUЄ
0001BD9280	FF	DF	EF	FF	00	00	CC	EE	F8	81	B0	B1	DD	27	20	64	яЯпя..МошЃ°±Э' d
0001BD9290	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
0001BD92A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
0001BD92B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
0001BD92C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....

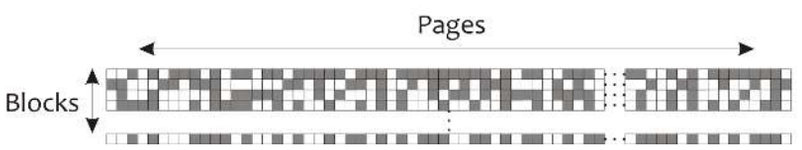
The whole hexadecimal array of bytes is divided by 16 bytes which is the industry standard.
The current address in hexadecimal format is displayed on the left, which means relative offset from the beginning of dump.
The byte address inside the line is displayed above with an accuracy to 1 byte.
In the center window there is user's data in hexadecimal representation.

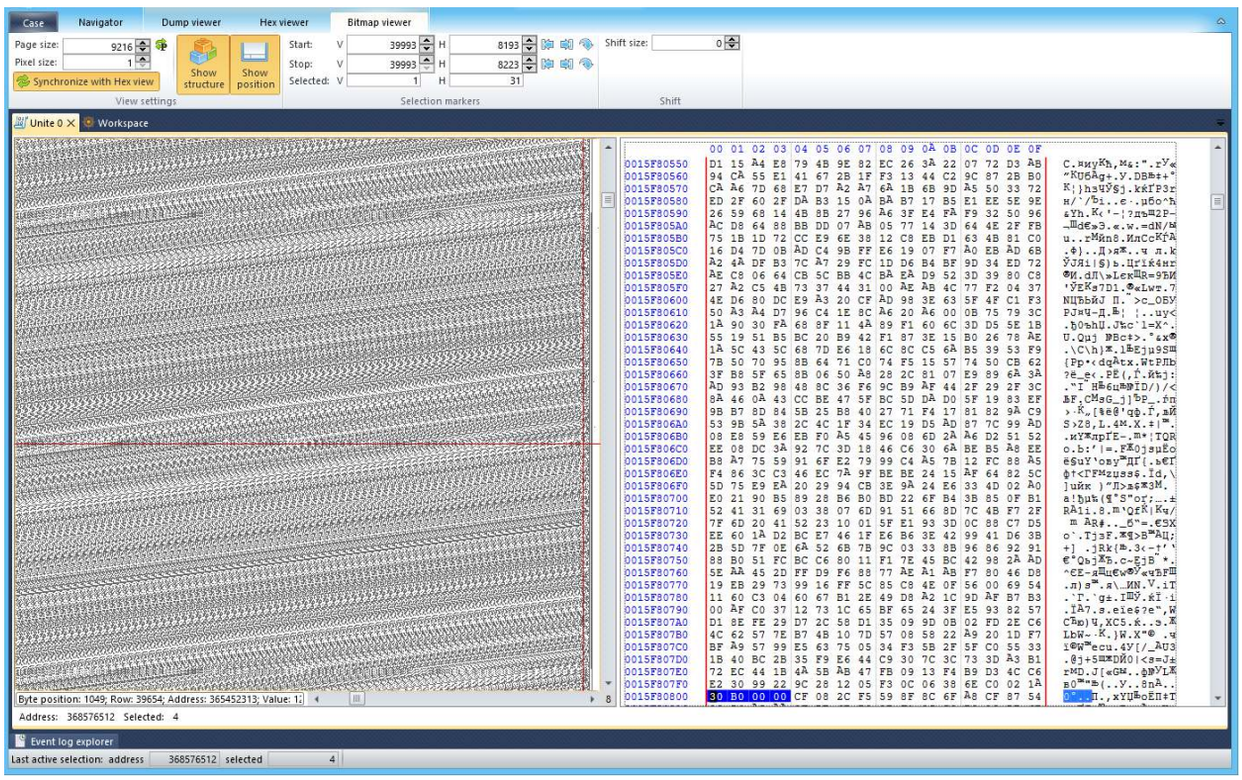


The Bitmap View mode is a tool for visualization of binary representation of dump. In this mode ~60-80% of dump analysis process takes place. It allows to analyze: data patterns, virtual block size, bad columns, ECC errors, XOR key, page structure, spare area structure, virtual block allocation scheme and many other patterns.

Transition from Hexadecimal to Bitmap data representation is performed in a following way







Toolbar with parameters of Bitmap mode is at in the upper part of the window.

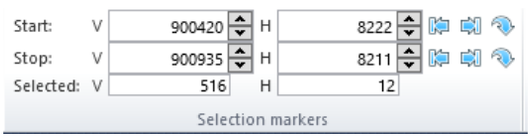
The content of dump in the visual binary representation is at the central part.

In the lower part of the window there is information about the current selection.

In Bitmap mode the pages are represented horizontally, the blocks vertically. This image representation shows the real physical structure of data in flash memory. The Bitmap mode is active, different measurements can be done there.

By pressing the mouse in the central part of the window where the binary content is, the selection tool is called, that allows to perform binary measurements. Pressing the left mouse button sets selection marker of beginning (red), pressing the right mouse button sets the end marker (violet). Measurements are made simultaneously - horizontally (inside of the page) and vertically (inside the block).

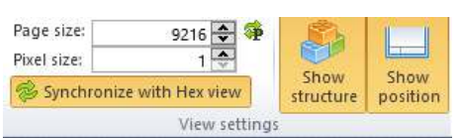
In the toolbar of Bitmap the selection markers show parameters of active selection.



Start V – red horizontal marker-line. It describes vertical beginning-coordinate of area from the beginning of dump.
Stop V – violet horizontal marker-line. It describes vertical end-coordinate of area from the beginning of dump.
Selected V describes the size of selected vertical area in pages (horizontal lines). This marker is used for determination of the virtual block size.

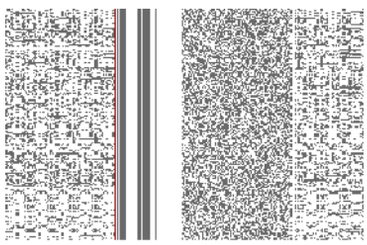
Start H – red vertical marker-line. It describes horizontal beginning-coordinate of area from the beginning of page.
Stop H – violet vertical marker-line. It describes horizontal end-coordinate of area from the beginning of page.
Selected H describes the size of selected horizontal area in bits (pixels within same line). This marker is used for analysis of page structure, determination of data and spare area size, bad columns analysis.

To use the Bitmap mode it's necessary to set correct page size (according to NAND configuration). In case if page size is set incorrectly, it shows garbage.

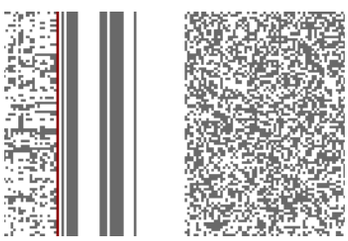


Pixel size determines how many pixels used to display one bit.

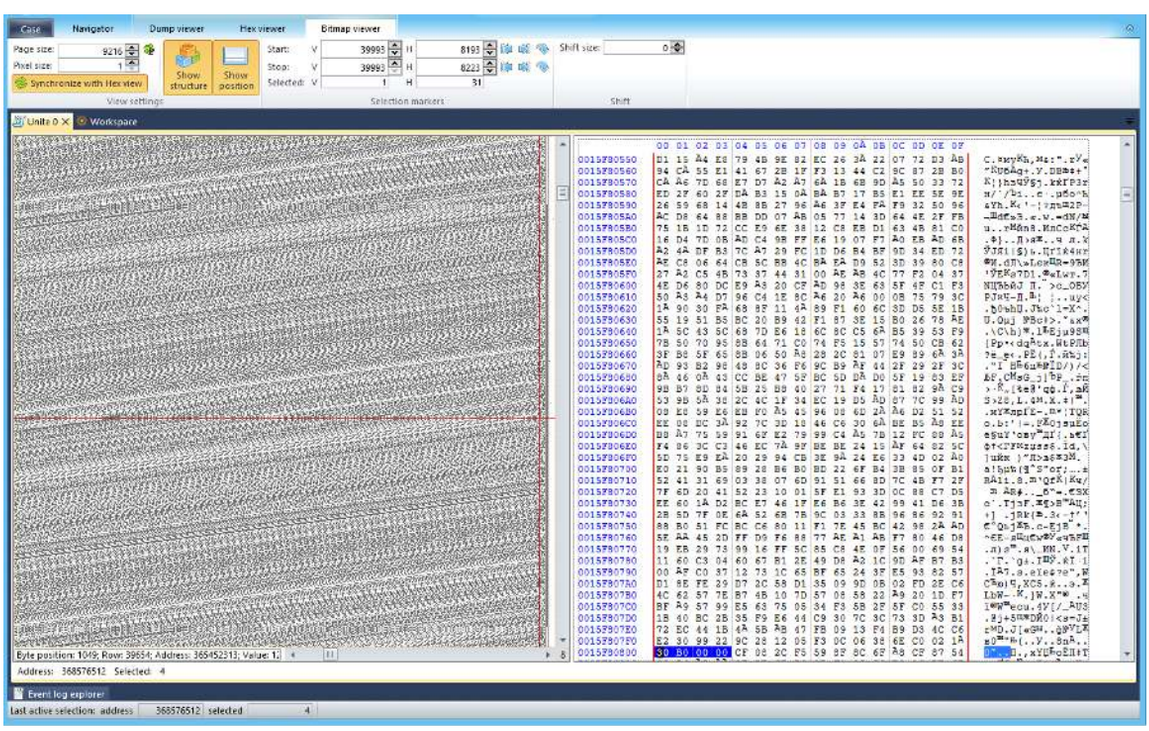
Set pixel size = 1 for analysis of page structure, virtual block size and bit errors.



Set pixel size = 2 for spare area and bad columns analysis.

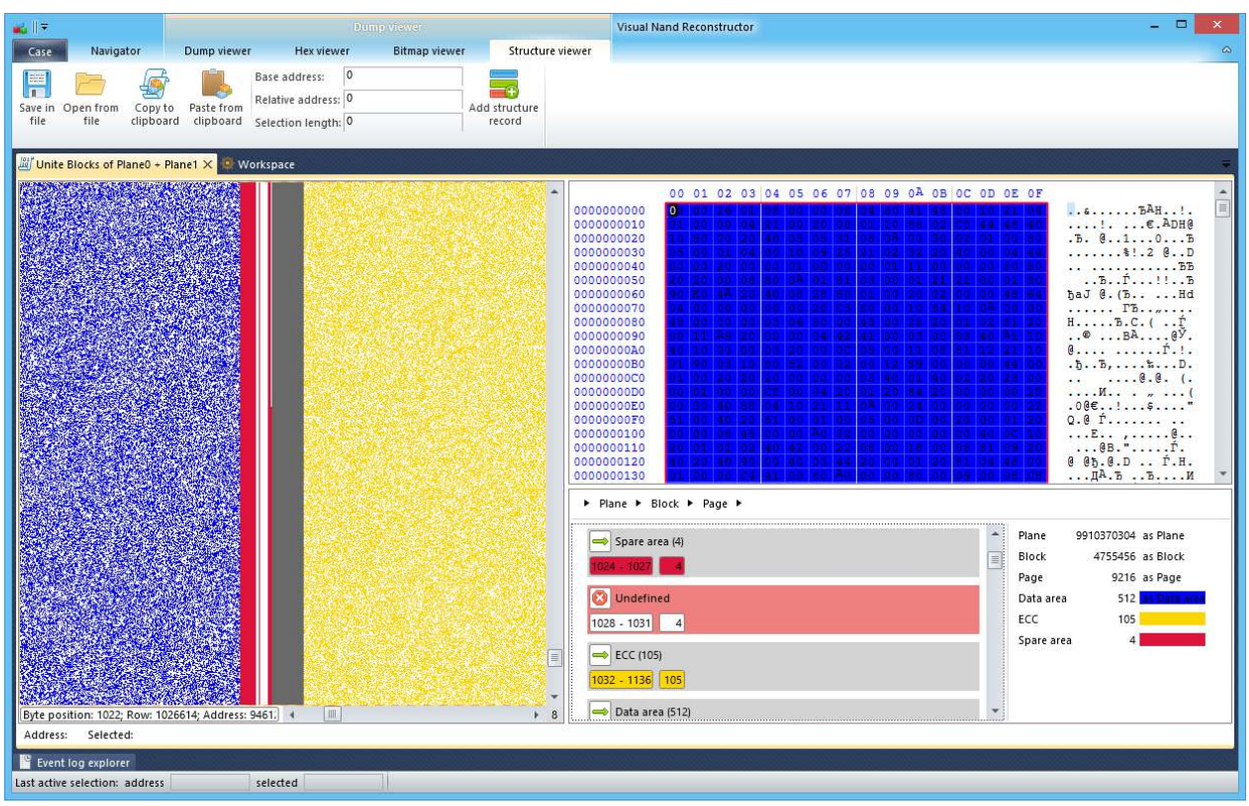


Synchronize with Hex View option allows to navigate through the data in two modes simultaneously – Hex View and Bitmap View.

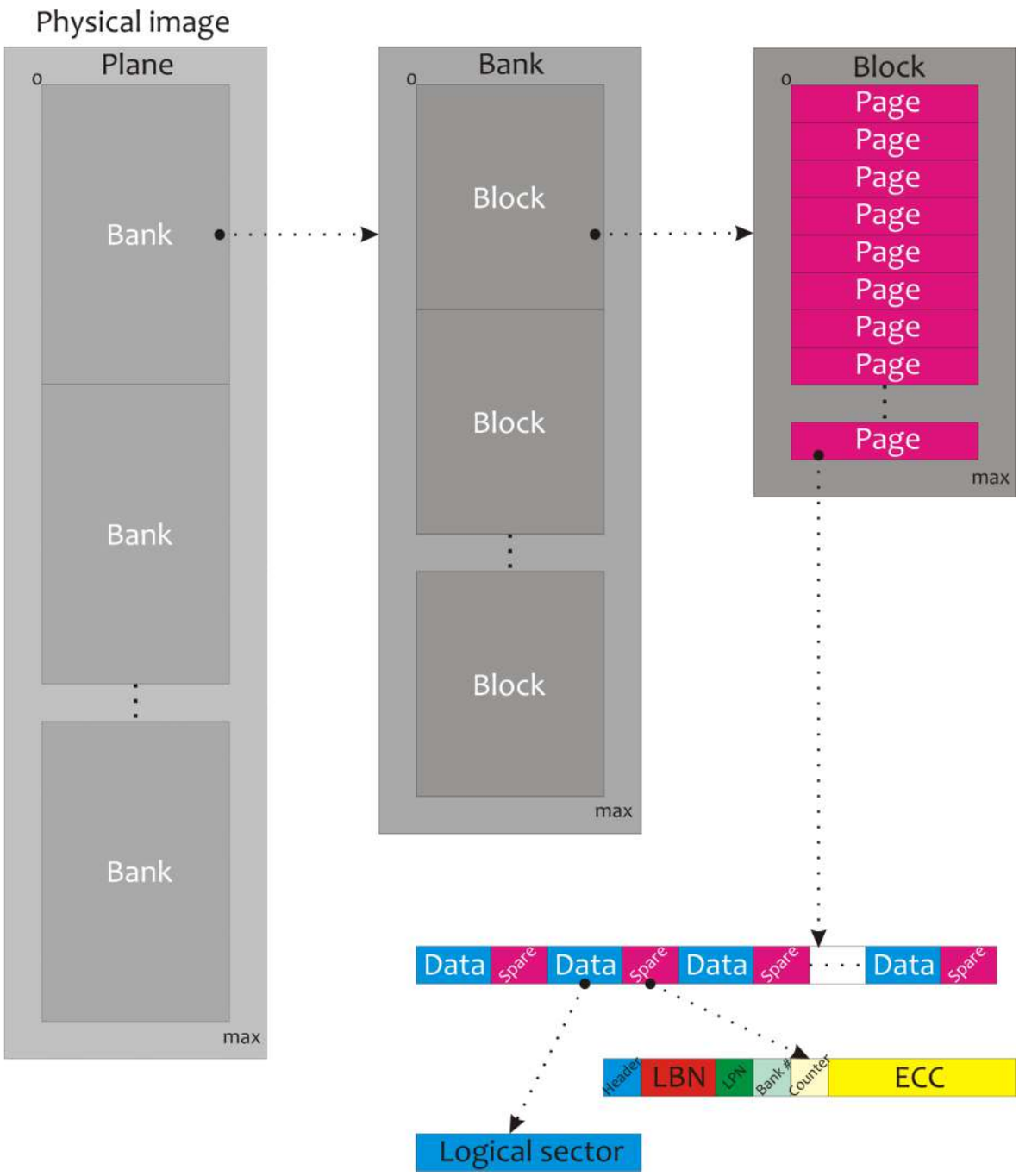


Different patterns of NAND chips and Bitmap usage can be found in the web article:
http://rusolut.com/binary-patterns-in-nand-flash-memory/

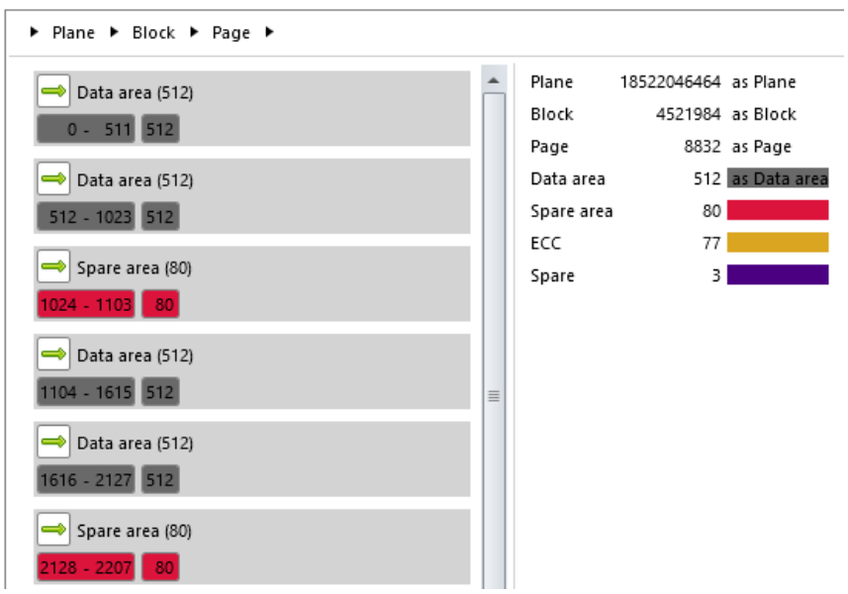
The Structure Viewer is a tool for the physical image structure description and visualization. Analysis and correct description of dump structure is the first and important step in the process of logical image reconstruction. To analyze the structure the Bitmap Viewer is used, and for describing structures and their sizes – Structure Viewer. All modes can be opened simultaneously in one window from Dump Viewer.



The physical image has multi-level structure. Each structure consists of sub-structures. Plane consists of Banks (in general, Plane = Bank). Plane/Bank consists of Blocks. Block consists of Pages. Page consists of Data area, Spare area and ECC area. Data area consists of bytes where user data (logical sectors) are stored, in modern flash devices its size is 1024b (sometimes 512b or 2048b). Spare area consists of Block header, LBN, ECC and some optional markers.



The main window of Structure Viewer consists of image structure tree (on the left) and structure library area (on the right).



When the chip is read, such structures as Plane, Block and Page are set automatically from NAND chip configuration.

The Plane is the parent dump structure and it's size equals to NAND crystal size (depending on number of planes per crystal, in general Crystal = Plane).

The Bank structure is a child stucture of Plane. Bank structure is not set by default, because most of modern controllers don't split physical space of NAND memory by banks and LBN numeration is solid. In case if the space of NAND memory is divided by banks, each bank would have numeration beginning with LBN 0000 and Plane should include the Bank structure. Number of banks usually equals power of 2 (1,2,4,8,16).

The Block structure is a child of plane/bank and it's default size is equal to the NAND chip's physical block. As the Virtual block may consist of several physical blocks, the size of this structure must be set in accordance with the virtual block size, which can be found using the Bitmap mode (1,2,4 physical blocks).

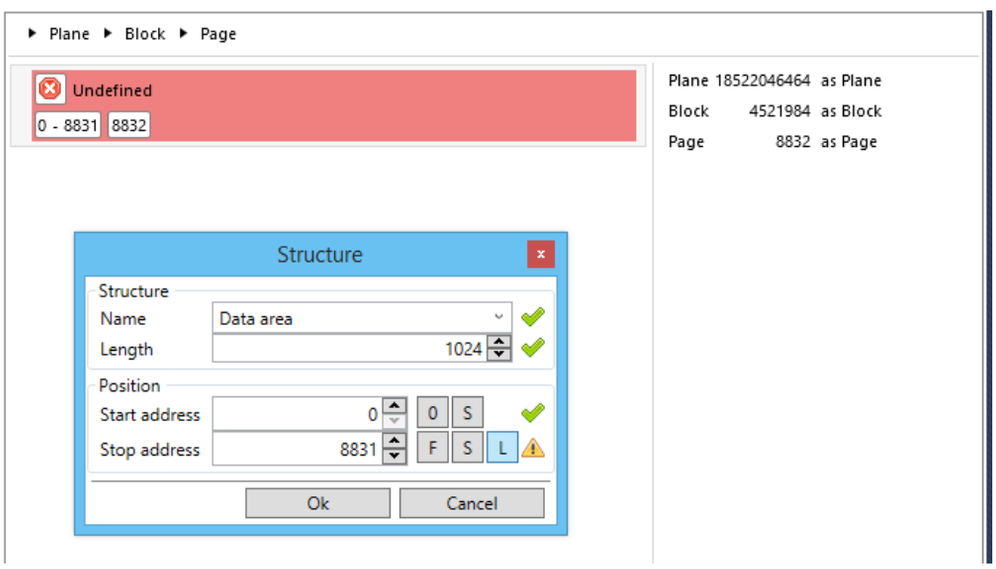
The Page structure is a child of block and it's size corresponds to physical size of page of NAND chip. This structure is defined during physical image extraction, in configuration window of NAND chip.

The Data Area structure is a child of page. The size of this structure can be found through the Bitmap mode while analysis of page structure. Usually, modern controllers use the Data area size equal to 1024 bytes (sometimes 512 or 2048 bytes). To set this structure, it must be manually "set as Data area" at structure library tab.

The Spare Area and ECC structures are children of page and set inside the page. Their size depends on page structure.

The LBN, Header and other structures are children of Spare Area and set with the aim of pointing parameters to the Markers Table for block arrangement, filtration and sorting for further logical image creation.

For each structure any color from the palette can be set, for visual simplification during analysis. It highlights structures in Hex and Bitmap modes. Every structure has beginning, end and size. To set any structure, it's necessary to determine it's parameters in Bitmap viewer.



To assign the structure it's necessary to double-click on undefined area and add name and length of structure. Also it's necessary to set position of structure within the parent area, it's start and end addresses. The special buttons used to set up addresses:

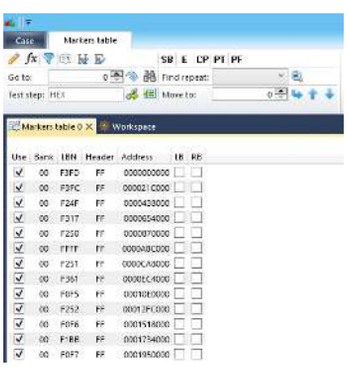
0 - start from zero
S - start/stop on selection marker (Bitmap mode selection markers)
F - full area (to the end of parent area)
L - length (Stop address = Start address + Length)

When the structure is set, it appears in structure library, at the right tab. To describe the page it's necessary to set Data area and Spare area once and then choose it from dropdown list.

Block list functions

The group Block list functions includes the tool for block sorting and filtering for logical image creation.

The Show Table function is used for block table displaying, analysis, filtration and sorting in accordance with LBN, Header etc.



Toolbar of block table has set of operations for block management algorithms.

Edit markers is used for appying inversion and mask to spare area markers.
Typical mask for LBN marker: not used; 0FFF; 07FF, 03FF.
Typical mask for Header: not used; F0.

Block filter is used for block filtering that won't transition into logical image (these blocks don't store user's data). The special syntax is used in block filter.

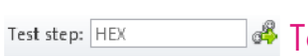
To filter by LBN range it's necessary to use syntax:
(xxxx-yyyy)
where xxxx is first block in sequence, yyyy is last block in sequence.
E.g. (1000-13FF).

To filter by Header it's necessary to use syntax:
xx/yy/zz
where xx,yy,zz are byte values that used in header to mark user's data blocks. E.g. 20/30.

The blocks can also be filtered manually, using flag ☑ near LBN.

Block sorting is used for block reordering according to their LBN in increasing order. In case when block addressing is independent for banks, blocks must be sorted by bank and by LBN inside each bank. This operation allows to set very flexible sorting rules for any controller configurations.

Find repeat is used for search of duplicated blocks that must be disabled before virtual to logical image transition. The duplicated blocks bring shifts into file system and file structure. They may be filtered by header, or manually by disabling flag ☑ near the block.



Test step is used for integrity control of the LBN sequence and analysis of places where sequence interrupts (Lost blocks). The test step should be set to 1/1 by default. The lost (missing) blocks bring shifts into file system and file structure. The dummy blocks must be added instead of lost blocks (click right button on block table).

When blocks re-ordered, the logical image can be created via Markers table element.

ruSolut
Data Investigation & Recovery Technology

Contact Us

Rusolut Sp. z o.o.

Address: 49 Kasprzaka st., Warsaw, Poland, 01-249

Phone: +48531999777
Email: info@rusolut.com
Web: www.rusolut.com

Useful links

Case samples
http://rusolut.com/case-samples/

Latest technology in our blog:
http://rusolut.com/blog/

F.A.Q.
http://rusolut.com/f-a-q/