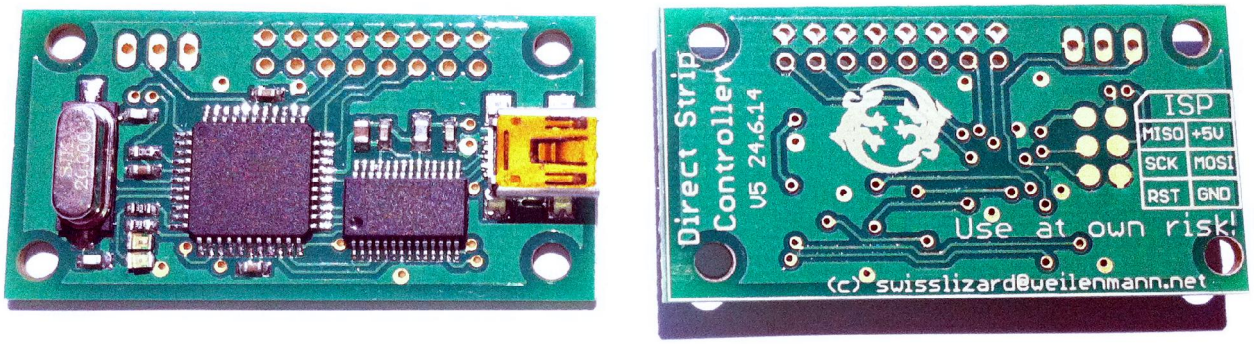
Direct Strip Controller



made for

by

Swisslizard

Table of Contents

Introduction

This small controller boards allows you to control led strips using WS2812 or WS2811 leds from your computer. The hardware is prepared to support other types of led strips as well, but the firmware still needs to be extended with some extra logic for those strips.

It consists of a USB chip which allows for fast communication with the PC and a Atmel micro controller which receives the data from the USB interface and generates the signal for the led strips.

Software

Driver

The controller board uses the FT245R chip from FTDI. To use the board you will have to install the appropriate driver, which is available for download on FTDIs page: http://www.ftdichip.com/Drivers/D2XX.htm
DOF needs only the D2XX drivers, but you might want to install the VCP drivers as well, since those are very easy to use if you want to write your own code.

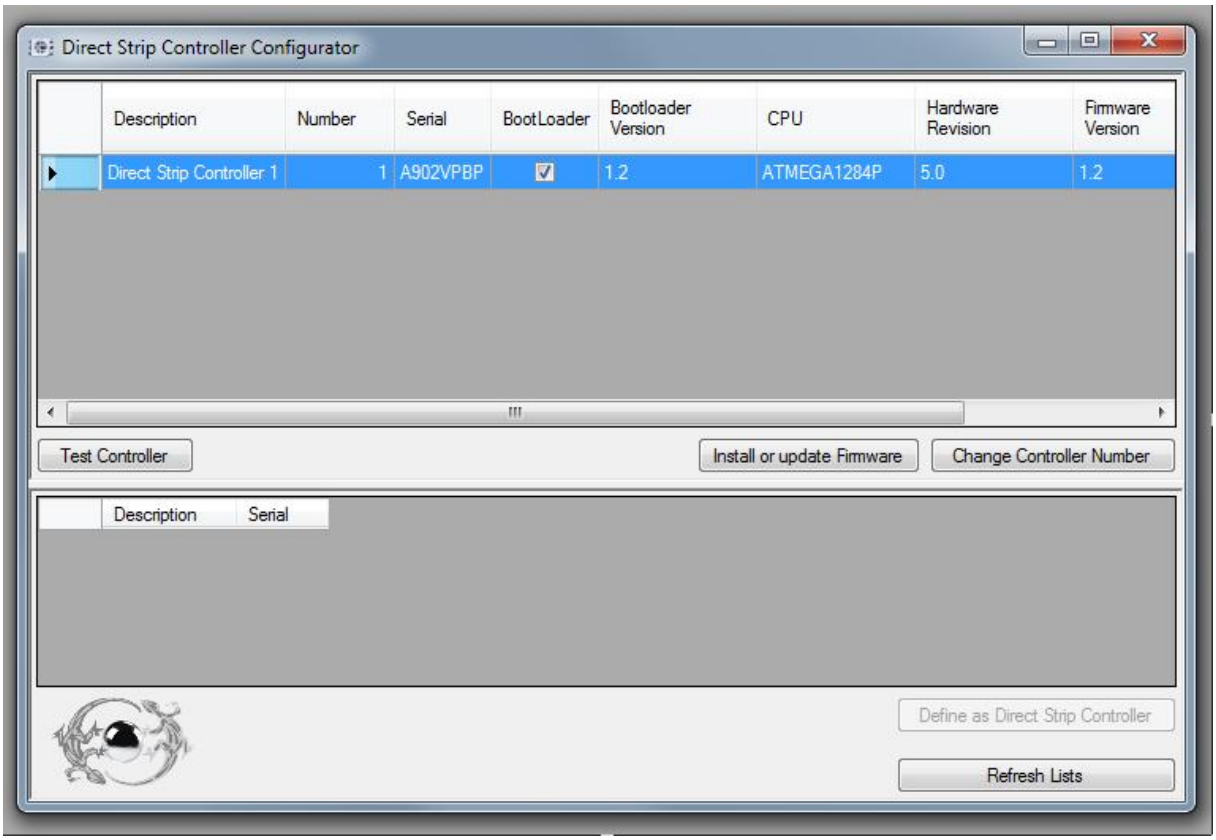
Firmware

The firmware has been developed using Atmel Studio 6, which is available from Atmels website. The firmware has been written in assembler to allow for max speed and precise cycle accurate output signals.

DirectStrip Controller Configurator

This small tool allows you to change the number of your controller board, update the firmware and to configure the settings of a new FT245 chip so it is recognized as a DirectStrip strip controller.

Main Window



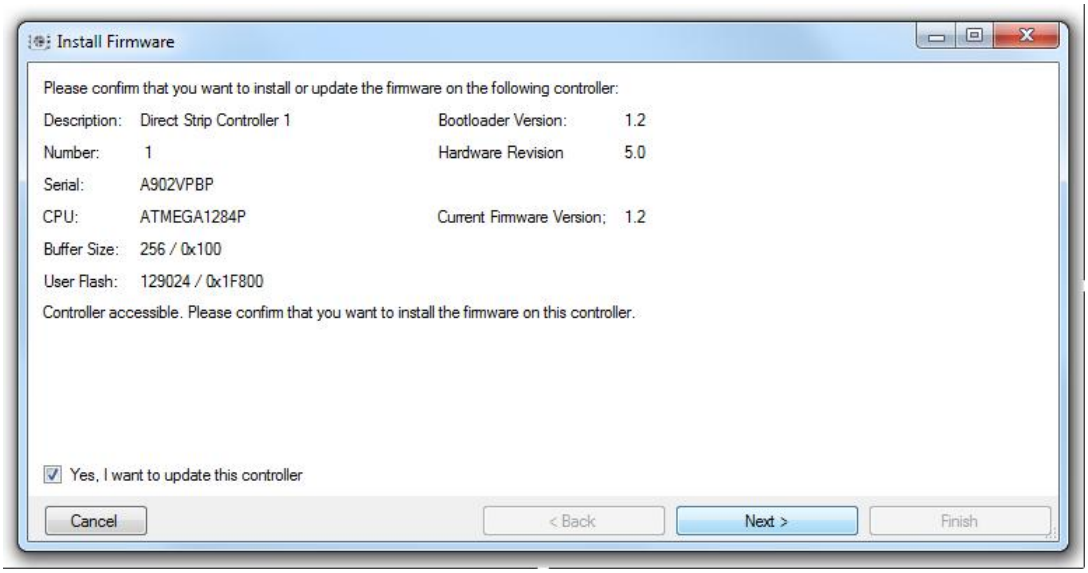
The upper list of the main window shows the currently detected DirectStrip controllers.
For controllers which have the bootloader installed, you can update the firmware, change the controller number and send some test patterns to a connected ledstrip.

The bottom list shows currently detected FT245 based units, including not configured DirectStrip controllers, but also other types of controllers which use the same type of hardware interface (e.g. Sainsmart). If you define a listed unit which is not a DirectStrip controller as a DirectStrip controller is might become unusable. So be sure you know what you are doing.

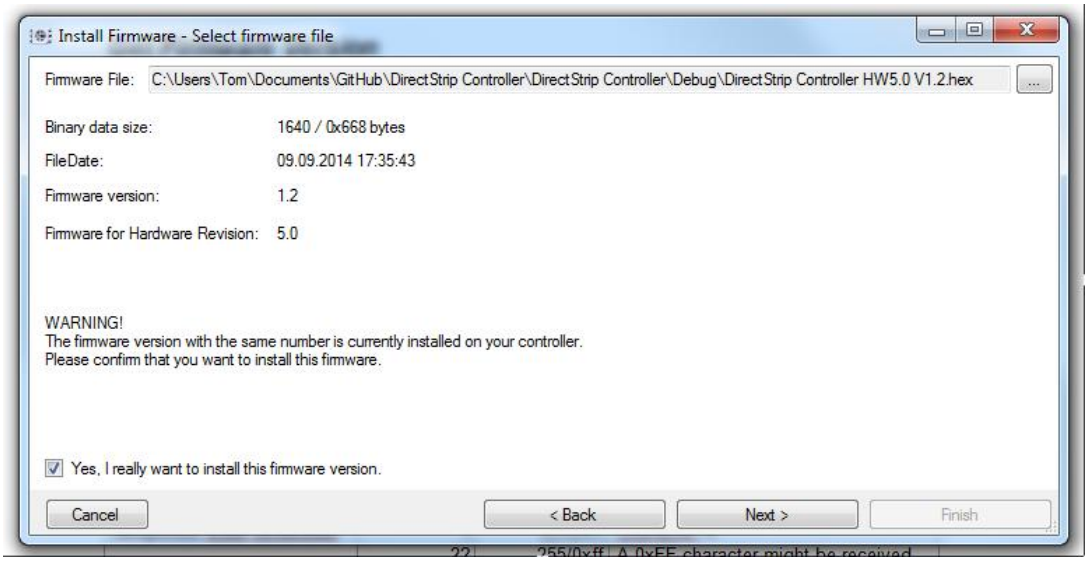
Update firmware

Warning!!!
Do not update your firmware for fun! The cpu of the controller has a limited number of write cycles and if something goes wrong during the firmware update your controller might become unusable.

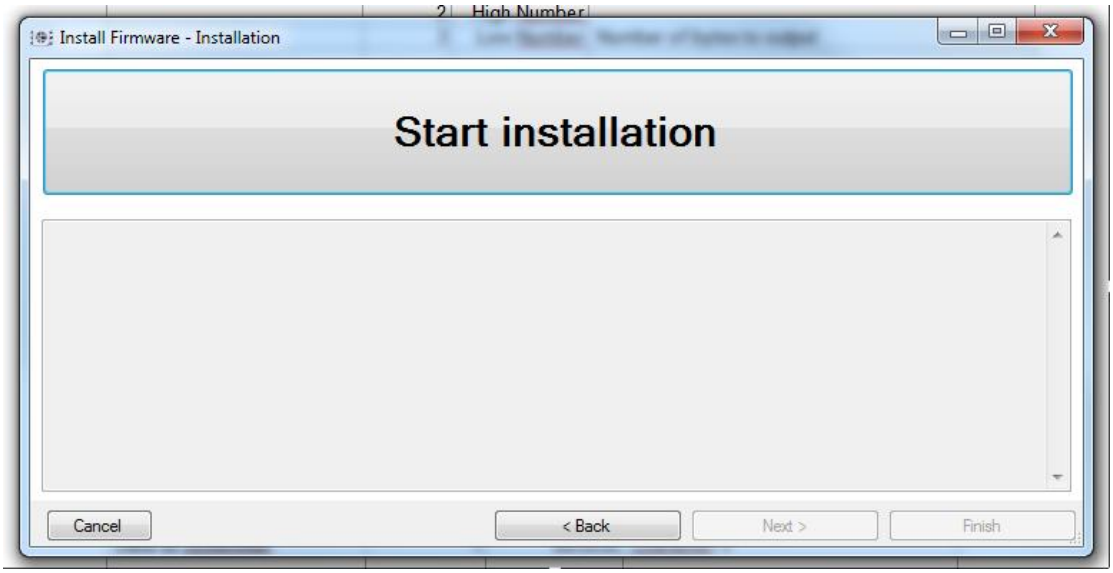
The firmware update wizard allows you to update the firmware of you controller board in three easy steps.



First you have to confirm the unit you want to update.



Second you choose the firmware you want to install (click the ... next to the Firmware File box). Depending on the firmware file you select you might need to check the extra confirmation block.



Third you start the installation of the firmware.
During the installation a progress bar and a log will keep you informed whats going on and whether the firmware installation has been successful or not.

Communication Protocol

The protocol of the controller board works in a simple serial fashion. Depending on the drivers you have installed, you can even access the board like a normal com port.

All commands for the controller start with a single character, specifying the command, followed by additional parameter bytes.
Every command is answered by the controller with single byte signalling acknowledge (A) or no acknowledge (N) of the last command.

Clear Data Buffer

Clears the data buffer on the controller.

	Byte Nr	Value	Remark
Data to controller	1	67/0x43	Character R
Response from controller	1	65/0x41	Character A

Receive Data

Tells the controller to receive a block of data and put it into the data buffer.
If you write your own software it is recommended that you choose a fast implementation when using this command, since the controller is able to receive data at high rates.

	Byte Nr	Value	Remark
Data to controller	1	82/0x52	Character R
	2	High Number	
	3	Low Number	Total number of data bytes to receive
	4	First data byte	
	5	Second data byte	

	….	More data bytes	
	Num Bytes-1	Second last data byte	
	Num Bytes	Last data byte	
Response from controller	1	65/0x41	Character A

Output Data

Sends data from the data buffer to the led strip. Normally this command will be used after the receive data command.

	Byte Nr	**Value**	**Remark**
Data to controller	1	79/0x4F	Character 0
	2	High Number	
	3	Low Number	Number of bytes to output
Response from controller	1	65/0x41	Ack is sent after all bytes have been sent to the led strip.

Get Firmware Version

Gets the version of the currently installed firmware.

	Byte Nr	**Value**	**Remark**
Data to controller	1	86/0x56	Character V
Response from controller	1	62/0x3E	Character >
	2	3/0x03	Number of answer bytes
	3	Version major	
	4	Version minor	0-99
	5	65/0x41	Ack

Start Bootloader

This commands starts the bootloader of the controller board by triggering a reset of the controller. Please take note that the controller doesn't receive any data for the next 3 seconds after the ack of the command.

	Byte Nr	**Value**	**Remark**
Data to controller	1	90/0x5A	Character Z
Response from controller	1	65/0x41	Character A
	2?	255/0xff	A 0xFF character might be received depending due to pin changes during the reset.

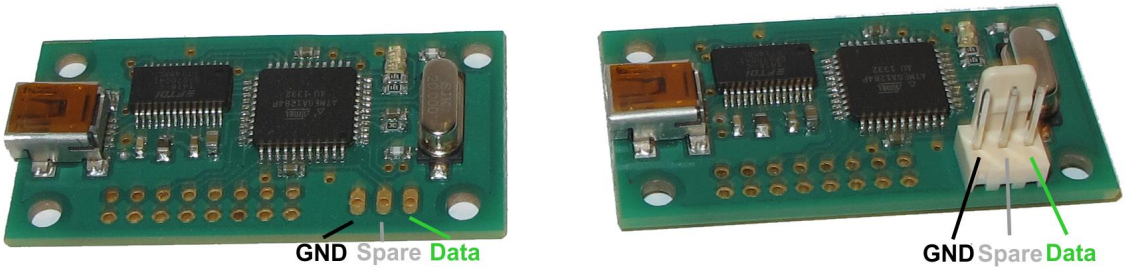
No acknowledge/Nack situations

If the above commands are used as expected, nack or no answer situations should at least in theory never occur. If you receive a nack anyway, there is likely a bug in your program, the firmware or less likely some problem receiving the data hardware wise. A simple procedure to react on a nack or missing answer is to send 0 bytes until another nack is received. After that second nack has been received the controller should be ready to receive the next command.

Wiring Instructions for Led Strips

Step 1: Connect led strips

Connect your WS2812 led strips to the controller board. You'll have to connect 2 wires: Data and Ground (GND). Both can be found on the strip connector of the controller board.



Warning!
Dont connect anything to the third pin marked Spare of the led strip connector. It is not yet active and reserved for possible future support of other types of led strips.

Step 2: Connect power supply

Connect your led strips to a suitable power supply. The controller board cant supply power for the led strips and doesn't have a power connector for that purpose.

Ledstrips can consume a lot of power. Typically a led strip based on WS2812 leds will consume up to 60ma (20ma for each color) per led, e.g. if you use 1 meter of strip with 60 leds you will need a power supply which can deliver at least 3.6A (60 * 60ma=3600ma).

For me power supplies from Meanwell have always worked well so far and are available in different strengths.

Voltage for the WS2812 led strips is 5 volts.

Remarks:

- If you use a long led strip, be sure to inject power not only at the start of the strip, but also after at least every 150-200 leds. Otherwise your strip will heat up to much, brightness and color of the leds change and operation will become unstable.
- The longer the cable from the power supply to the led strip gets and the longer the led strip is, the thicker the cable should be.
- Always keep in mind that those strips operate on 5volts only and that the currents (Ampere/A) required can be quite high. High currents can develop a lot of heat if you dont do your wiring properly!

Step 3: Connect USB

Connect the controller board to your computer using a normal USB cable.

To start using the board check the chapter on the software in drivers.

DirectOutput framework configuration

Cabinet config file

To use this controller board with the DirectOutput framework (DOF) you will need a cabinet config file. The cabinet config file is a XML file with the following minimal structure:

```
<?xml version="1.0"?>
<Cabinet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
	<Name>Name of the cabinet</Name>
	<OutputControllers>
		... configuration of any number of output controllers ...
	</OutputControllers>
	<Toys>
	... configuration of any number toys ...
	</Toys>
</Cabinet>
```

To configure the controller board for DOF you need to specify a DirectStripController in the OutputControllers section, one or several LedStrip toys and a LezWizEquivalent toy in the Toys section.

```
<?xml version="1.0"?>
<Cabinet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
	<Name>Name of the cabinet</Name>
	<OutputControllers>
		<DirectStripController>
			<Name>ControllerName</Name>
			<ControllerNumber>1</ControllerNumber>
			<NumberOfLeds>235</NumberOfLeds>
		</DirectStripController>
	</OutputControllers>
	<Toys>
		<LedStrip>
			<Name>BackBoard</Name>
			<Width>32</Width>
			<Height>3</Height>
			<LedStripArrangement>LeftRightTopDown</LedStripArrangement>
			<ColorOrder>GRB</ColorOrder>
			<FirstLedNumber>1</FirstLedNumber>
			<FadingCurveName>SwissLizardsLedCurve</FadingCurveName>
			<OutputControllerName>ControllerName</OutputControllerName>
		</LedStrip>
		<LedStrip>
			<Name>SideboardRight</Name>
			<Width>1</Width>
			<Height>65</Height>
			<LedStripArrangement>TopDownLeftRight</LedStripArrangement>
			<ColorOrder>GRB</ColorOrder>
			<FirstLedNumber>97</FirstLedNumber>
			<FadingCurveName>SwissLizardsLedCurve</FadingCurveName>
			<OutputControllerName>ControllerName</OutputControllerName>
		</LedStrip>
		<LedStrip>
			<Name>SideboardLeft</Name>
			<Width>1</Width>
			<Height>65</Height>
			<LedStripArrangement>TopDownLeftRight</LedStripArrangement>
			<ColorOrder>GRB</ColorOrder>
			<FirstLedNumber>167</FirstLedNumber>
			<FadingCurveName>SwissLizardsLedCurve</FadingCurveName>
			<OutputControllerName>ControllerName</OutputControllerName>
		</LedStrip>
		<LedWizEquivalent>
			<Name>LedWizEquivalent50</Name>
			<LedWizNumber>50</LedWizNumber>
			<Outputs>
				<LedWizEquivalentOutput>
```

```
                                        <OutputName>BackBoard</OutputName>
                                  <LedWizEquivalentOutputNumber>1</LedWizEquivalentOutputNumber>
                                </LedWizEquivalentOutput>
                                <LedWizEquivalentOutput>
                                        <OutputName>SideboardRight</OutputName>

        <LedWizEquivalentOutputNumber>4</LedWizEquivalentOutputNumber>
                                </LedWizEquivalentOutput>
                                <LedWizEquivalentOutput>
                                        <OutputName>SideboardLeft</OutputName>

        <LedWizEquivalentOutputNumber>10</LedWizEquivalentOutputNumber>
                                </LedWizEquivalentOutput>
                        </Outputs>
                </LedWizEquivalent>
        </Toys>
</Cabinet>
```

DirectStripController section

The following section defines the output controller:

```
 <DirectStripController>
        <Name>ControllerName</Name>
        <ControllerNumber>1</ControllerNumber>
        <NumberOfLeds>235</NumberOfLeds>
 </DirectStripController>
```

- Name can be anything you like.
- ControllerNumber is the number of your controller board. You can use the tool supplied with the board to change the controller number of your boards.
- NumberOfLeds is the total number of RGB leds on your ledstripe. This value defines how much data DOF sends to the controller board and how much memory it reserves for the led strip data.

LedStrip toy sections

The LedStrip toy sections look a follows:

```
<LedStrip>
        <Name>BackBoard</Name>
        <Width>32</Width>
        <Height>3</Height>
        <LedStripArrangement>LeftRightTopDown</LedStripArrangement>
        <ColorOrder>GRB</ColorOrder>
        <FirstLedNumber>1</FirstLedNumber>
        <FadingCurveName>SwissLizardsLedCurve</FadingCurveName>
        <OutputControllerName>ControllerName</OutputControllerName>
</LedStrip>
```

- Name can be anything you like.
- FirstLedNumber is the number of the first led of the strip which is part of the toy.
- OutputControllerName is the name of the output controller which you have defined in the DirectStripController section.
- Width and Height define the size of your led stripe arrangement. This allows you to define a matrix of leds. If you use just a single strip in your arrangement, set these values based on the main direction of your ledstrip, e.g. width: 32, height: 1 for a single strip on a backboard or width:1, height:65 for a single strip on a sideboard of your cabinet. If you have a matrix set width and height to the number of leds in x resp y direction.
- LedStripArrangement is used to tell DOF how the ledstrips in your matrix are wired. If your arrangement consist of a single strip this settings doesn't have any effect. The following values are supported:

    - LeftRightTopDown
    - LeftRightBottomUp
    - RightLeftTopDown
    - RightLeftBottomUp
    - TopDownLeftRight
    - TopDownRightLeft
    - BottomUpLeftRight
    - BottomUpRightLeft
    - LeftRightAlternateTopDown
    - LeftRightAlternateBottomUp
    - RightLeftAlternateTopDown
    - RightLeftAlternateBottomUp
    - TopDownAlternateLeftRight
    - TopDownAlternateRightLeft
    - BottomUpAlternateLeftRight
    - BottomUpAlternateRightLeft
- ColorOrder defines the order of colors of the RGB leds on your led strip. The following values are supported:
    - RGB = Red-Green-Blue (usual color order)
    - RBG = Red - Blue - Green
    - GRB = Green - Red - Blue (WS2812 led chips are using the scheme)
    - WS2812 = WS2812 led chip (uses green - red - blue color order)
    - GBR = Green - Blue - Red
    - BRG = Green - Blue - Red
    - BGR = Blue - Green - Red
- FadingCurveName allows you to define which fading cruve to use for your leds. Fading curves can be used to adjust the brightness of the leds to the brigthness perception of the human eye or also to achieve special effects. DOF has a number of predefined fading curves, but user defined curves (see DOF docu for details) can be used as well. The following predefined curves are available:
    - Linear = A linear curve, where each element will map to a value which is equal to the element index (0=0, 1=1 .... 254=254,255=255).
    - Linear0To224 = A linear curve, which maps 0 to 255 value range into a new range of 0 to 224.
    - Linear0To192 = A linear curve, which maps 0 to 255 value range into a new range of 0 to 192.
    - Linear0To160 = A linear curve, which maps 0 to 255 value range into a new range of 0 to 160.
    - Linear0To128 = A linear curve, which maps 0 to 255 value range into a new range of 0 to 128.
    - Linear0To96 = A linear curve, which maps 0 to 255 value range into a new range of 0 to 96.
    - Linear0To64 = A linear curve, which maps 0 to 255 value range into a new range of 0 to 64.
    - Linear0To32 = A linear curve, which maps 0 to 255 value range into a new range of 0 to 32.
    - Linear0To16 = A linear curve, which maps 0 to 255 value range into a new range of 0 to 16.
    - InvertedLinear = This is a inverted linear curve where 255=0, 254=1 and so on until 1=254 and 0=255.

- SwissLizardsLedCurve = A fading curve for leds defined by SwissLizard. This curve is not fully correct when it comes to theoretically needed mapping values, but it is some kind of compromise between possible value range and the desired values.

LedWizEquivalent section

The LedWizEquivalent section allows DOF to map effect settings from ini files with table configurations to the led strips.

```
<LedWizEquivalent>
	<Name>LedWizEquivalent50</Name>
	<LedWizNumber>50</LedWizNumber>
	<Outputs>
		<LedWizEquivalentOutput>
			<OutputName>BackBoard</OutputName>
			<LedWizEquivalentOutputNumber>1</LedWizEquivalentOutputNumber>
		</LedWizEquivalentOutput>
		<LedWizEquivalentOutput>
			<OutputName>SideboardRight</OutputName>
			<LedWizEquivalentOutputNumber>4</LedWizEquivalentOutputNumber>
		</LedWizEquivalentOutput>
		<LedWizEquivalentOutput>
			<OutputName>SideboardLeft</OutputName>
			<LedWizEquivalentOutputNumber>10</LedWizEquivalentOutputNumber>
		</LedWizEquivalentOutput>
	</Outputs>
</LedWizEquivalent>
```

- Name can be anything you like.
- LedWizNumber is the number of the ini file to be associated with the LedWizEquivalent toy.
- The output section defines which column of the ini file is mapped to which led strip toy:
  - OutputName has to specify the name of the LedStrip toy to be controlled. Take note that this is different from normal OutputName settings for LedWizEquivalent toys, where the name of a output is defined.
  - LedWizEquivalentOutputNumber is column of the ini file which is to be associated with the led strip specified in OutputName.

Table configurations

The led strips connected to the contoller board can be controlled by ini files like other toy.

The same parameters as for other toys are accepted, but the are some additional paras for toys consisting of more than one element.

Trigger parameters

The first part of a setting defines how the setting/effect is triggered and must always be one of the following:

- **TableElementTypeChar plus Number** (e.g. S48 for solenoid 48) determines which table element is controlling the specified effect.
- **List of TableElementTypeChars plus Numbers** delimited by | (e.g. S48|W12|L59). This setting assigns the same effect to all table elements in the list.
- **Condition** which controlles wether the effect is triggered or not. Conditions must always be in brackets. Example: (S48=1 and W29=0 and (L59=1 or L43<>0)).
- **On** resp. **1** turns the specified effect constantly on.
- **B** defines a static (not externaly controlled) blinking.

General parameters

The second and following parts of a setting can contain one or several of the following paramaters:

- **Color name** as specified in the colors section of the file. Only valid as the second value (e.g. S48 Blue).
- **Hex color definition** (e.g. #ff0000ff for opaque red). Take note that these color definitions allow for values from 0-255 in contrast to the colors section which only support 0-48. Hex color definitions can contain 3 or 4 parts (without or with alpha value). Setting is only accepted as the second value.
- **Blink** defines blinking with a default interval of 500ms.
- **I{Number}** defines a intensity/level of the output. The number can either be specified as a decmal number between 0 (e.g. I0 for off) and 48 (e.g. I48 for fully on) or as a hexadecimal number between 00 (off) and FF (fully on) with a leading # (e.g. I#80 for 50% power). This settings does only have a effect for settings without a color definition.
- **L{Number}** defines the layer on which the setting operates. In most cases the setting is not required, since DOF will assign ascending layer numbers to the settings for a column anyway.
- **W{NumberOfMilliseconds}** defines a wait period resp. delay before the effect executes after it has been triggered.
- **M{NumberOfMilliseconds}** defines the minimum duration for the effect in milliseconds.
- **Max{NummberOfMilliseconds}** defines the maximum duration for the effect in milliseconds.
- **F{NumberOfMilliseconds}, FU{NumberOfMilliseconds}, FD{NumberOfMilliseconds}** are used to specify the fading duration in

milliseconds. *F* sets the duration for both fading up and down, *FU* controls fading up only and *FD* fading down only.
- **E{NumberOfMilliseconds}** specifies a extended duration in milliseconds for the effect (after it has been turned off).
- **BL{Number}** specifies the value of the blink effect during the low period of the blinking (High value=trigger value of the effect, typicaly 255).
- **BPW{Percentage}** defines the blink pulse width in percent of the blink interval. Valid values are 1-99, default value if not defined is 50.
- **BNP{NumberOfMilliseconds}** defines the interval (duration of one on/off period) for nested blinking. This allows to define a second level of blinking within the *on* period of the normal blinking.
- **BNPW{Percentage}** defines the blink pulse width for nested blinking in percent of the blink interval.
- **Invert** inverts the effect, so the effect will be active when it is normaly inactive and vice versa.
- **NoBool** indicates that the trigger value off the effect is not to be treated as a boolean value resp. that the daufault mapping of the value to 0 or 255 (255 for all values which are not 0) should not take place. The number can either be specified as a decmal number between 0 and 48 (e.g. BL3) or as a hexadecimal number between 00 and FF with a leading # (e.g. BL#30).
- **Numeric Values** without any extra character can be used to specify the duration of the effect or the blinking behaviour. If blinking has been defined, one or two numeric values are parsed. Value 1 controls the blink interval in milliseconds, while value 2 defines the number of blinks. If no blinking has been defined, only one numeric values which is used to defined the duration of the effect in milliseconds is parsed.

Matrix/area effect parameters

For adressable ledstrips and other toys which implement the IMatrixToy interface the following extra parameters can be used to control the hardware referenced by the matrix. For settings controlling a matrix you have to use at least one of these paras, so DOF realizes that a matrix/area is to be controlled.

The matrix effects and parameters can be combined with the general paras mentioned above.

General Matrix Paras

The following 4 paramaters are specifying the area of a matrix which is to be influenced by a matrix effect:

- **AL{LeftPosition}** defines the left of the upper left corner of the area of the matrix which is to be controlled by the effect. Position is expressed in percent of the matrix width (0-100).
- **AT{TopPosition}** defines the upper part of the upper left corner of the area of the matrix which is to be controlled by the effect. Position is expressed in percent of the matrix height (0-100).
- **AW{Width}** defines the width of the area of the matrix which is to be controlled by the effect. Width is expressed in percent of the matrix width (0-100).

- **AH{Height}** defines the height of the area of the matrix which is to be controlled by the effect. Height is expressed in percent of the matrix height (0-100).

Shift Effect Paras

The matrix shift effect moves a color/value with a defineable direction, speed and acceleration through the matrix:

- **AD{DirectionCharacter}** defines the direction for area effects having a direction parameter (e.g. the ColorShiftEffect). Valid directions are: R-Right, L-Left, U-Up, D-Down.
- **AS{Speed}** defines the speed for matrix effects have a speed parameter (e.g. the ColorShiftEffect) expressed in percent of the effect area per second. 100 will shift through the effect area in one second, 400 will shift through the effect area in 1/4 second. Min. speed is 1, max. speed is 10000.
- **AA{Acceleration}** defines the acceleration of the speed for matrix effects (e.g. ColorShiftEffect), expressed in percent of the effect area per second. Acceleration can be positive (speed increases) or negative (speed decreases). Speed will never decrease below 1 and never increase above 10000.

Flicker Effect Paras

The flicker effect generates random flickering with a defineable density and duration for the single flickers:

- **AFDEN{Percentage}** defines the density for the flicker effect. Density is express in percent a has a valid value range of 1 to 99.
- **AFMIN{DurationInMilliseconds}** defines the min duration for the flicker of a single led in milliseconds.
- **AFMAX{DurationInMilliseconds}** defines the max duration for the flicker of a single led in milliseconds.

Bitmap Effect Paras

DOF can display a part of a bitmap image on a matrix toy. The defined part of the bitmap is scaled to the size of the matrix, so the actual resolution of the matrix does not matter. If you specify a bitmap effect by using one of the following parameters, DOF will try to load a bitmap image (gif, png, jpg and more should all work) from the same directory as the ini file. The bitmap image has to be named like the short rom name in the first collumn of the ini file (e.g. mm.png for Medival Madness or afm.gif for Attack from Mars).

- **ABL{LeftPostionInPixels}** defines the left/horizontal part of the upper left corner of the part of the bitmap to be displayed. Defaults to 0 if not specified.
- **ABW{WidthInPixels}** defines the width of the part of the bitmap to be displayed. Defaults to the total width of the image if not specified.
- **ABT{TopPositionInPixels}** defines the upper/vertical part of the upper left corner of the part of the bitmap to be displayed. Defaults to 0 if not specified.
- **ABH{HeightInPixels}** specifies the height of the part of the bitmap to be displayed. Defaults to the total height of the image if not specified.
- **ABF{FrameNumber}** indicates the frame of the image to be displayed. This setting is only relevant if you use animated gif images. Defaults to the first frame of the animated gif if not specified.

Bitmap Animation Paras

The following extra paras can be used in addition to the bitmap paras to animate the bitmap display on the matrix:

- **AAC{CountOfFrames}** specifies the total number of frames of the animation.
- **AAF{FramesPerSecond}** specifies the number of frames per second.
- **AAD{FrameExtractionDirectionCharacter}** defines the direction in which DOF moves through the source image to generate the animation. Valid values for the direction character are L=Left, D=Down, F=Frame (for animated gifs)
- **AAS{FrameExtrationStepSize}** defines the size of the steps DOF takes to move through the source image to generate the animation. The step size is either in pixels or in frames.
- **AAB{AnimationBehaviourCharacter}** defines the behaviour when the animation is triggered. Valid values are O=show animation once, L=Start at beginning and show animation in a loop (default), C=Continue at last position and show animation in a loop

The following image might give a a better idea what these parameters do. It shows the behaviour for a setting like S48 AL0 AT10 AW100 AH20 AAC116 AADD AAS5 AAF30 AABL.