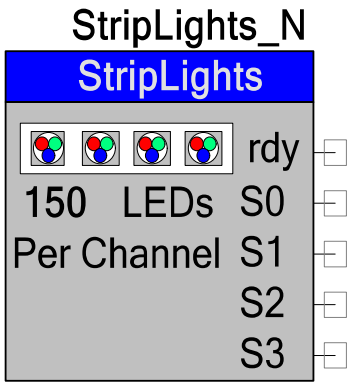
StripLights (NeoPixels)

2.2

Features

- Drives popular WS2811/12 LEDs
- Drives 1 to 16 strings of LEDs
- Supports 24-bit RGB and Color Lookup Table
- Supports 5x7 character font
- LEDs arranged as NxM graphics array
- Pixel, line, rectangle, circle primitives supported
- Triggered refresh occurs in ISR
- 800 kHz and 400 kHz controllers supported



General Description

The StripLights, also known as Neopixels, component will control up to 3000 RGB LEDs with a single PSoC 4A (CY8C4345). The user may configure the LEDs as one long string or up to 16 rows of strings to create a dot matrix display. Simple commands are provided to draw pixels, lines, rectangles, circles, and characters. The user uses the configuration dialog to set the number of strings and LEDs per string to create the array. Figure 1 below shows examples of how the symbol expands as the number of strings is changed between 1 and 16 strings of LEDs.

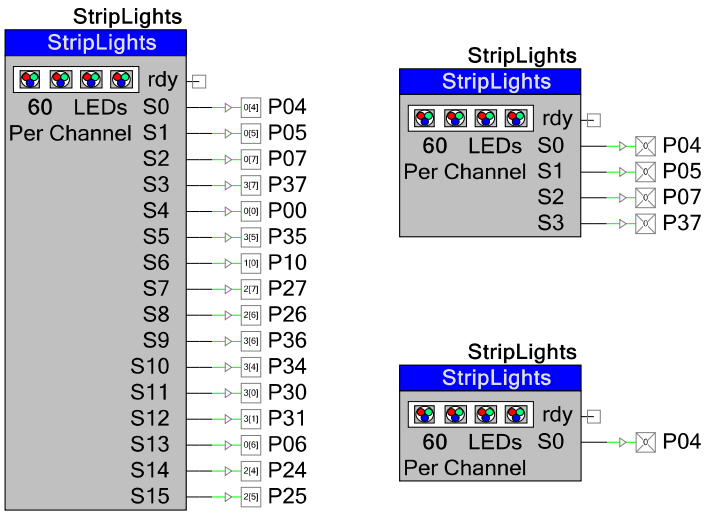


Figure 1 Examples project schematic

Input/Output Connections

This section describes the various input and output connections for the StripLights. An asterisk (*) in the list of I/Os indicates that the I/O may be hidden on the symbol under the conditions listed in the description of that I/O.

rdy – Digital Output

The “rdy” signal goes high when the transfer to a single string is complete.

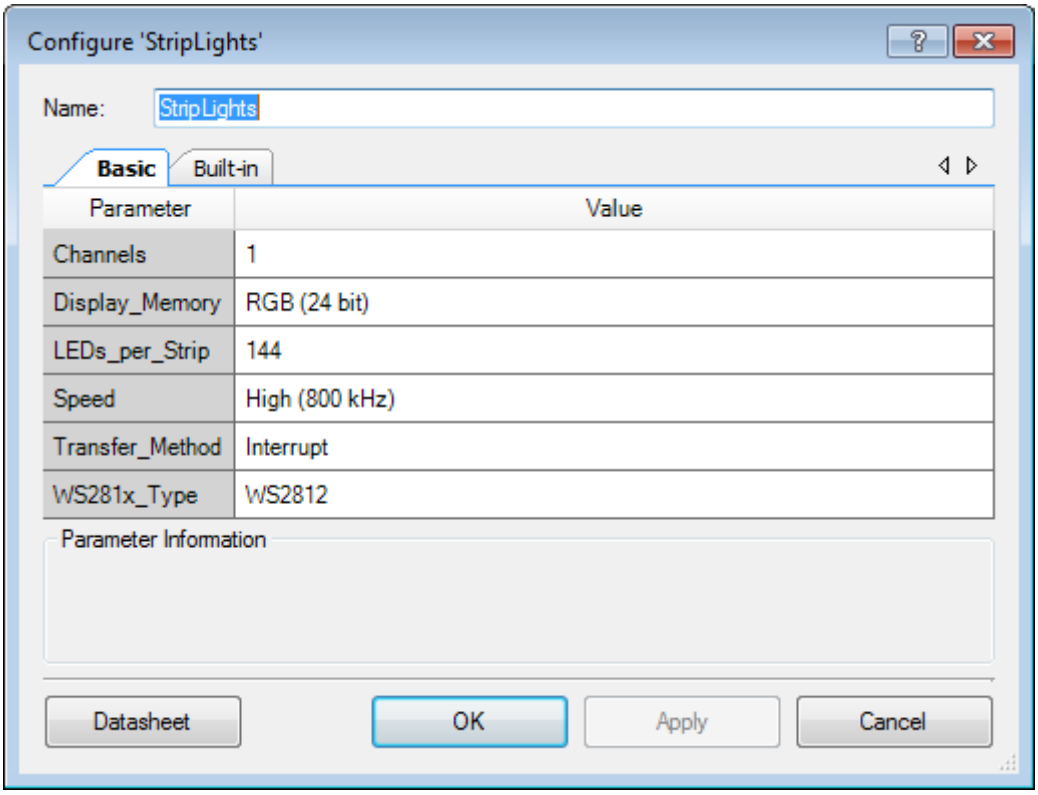
S0-15 – Digital Output

There can be between 1 and 16 outputs depending on the configuration of the component. For best results, place the LED strips in sequential order from S0 to Sn.

Parameters and Setup

Drag a StripLights component onto your design and double-click it to open the Configure dialog.

Figure 2 Configure StripLights Dialog



The StripLights component provides the following parameters.

Channels

Set this parameter to the number of LCD strings connected to the PSoC. The number of channels defines the maximum Y-size of the memory array.

Display_Memory

This parameter determines if the color information is stored as a 32-bit RGB value or an 8-bit index to a color lookup table. The RGB (24-bit) method generates faster code, but limits the maximum LEDs driven to about 700. The LUT (8-bit) mode is not as code efficient, but will support about 3000 RGB LEDs.

- LUT (8-bit)
- RGB (24-bit)

LEDs_per_Strip

This parameter is set to the number of LEDs per strip or channel. It sets the maximum X-size in the memory array.

Example:

If the Channels is set to 6 and the LEDs_per_Strip is 8, the usable memory graphics area (x,y) is LEDs_per_Strip x Channels or (8,6). The actual X-range is 0 to 7, and the actual Y-range is 0 to 5.

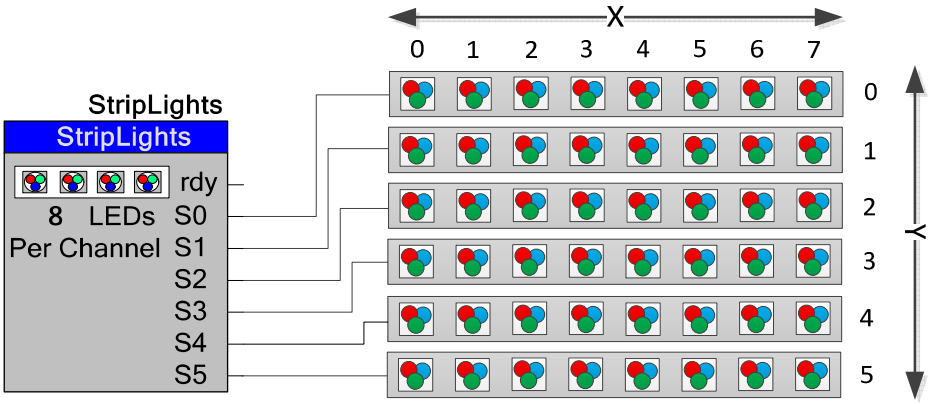


Figure 3 LED Strip configuration

Speed

This WS2811/12 parts operate at one of two frequencies, 400 kHz or 800 kHz, select the speed that is specified by the manufacturer. All parts must be the same speed.

Transfer_Method

This parameter sets the transfer type. Currently only the Interrupt method is supported. A future version will support DMA on PSoC 5LP only.

- Firmware
- **Interrupt (Recommended)**
- DMA (Not supported at this time.)

WS281x_Type

This parameter sets the device as either WS2811 or WS2812. Although these devices are very similar, the color data order is different. The WS2812 option will also support the new 4 pin WS2812b parts as well.

Resources

The StripLights component requires between 2 and 4 UDBs depending on the channel count.

Flash: TBD

SRAM: One byte per RGB LED in LUT(8-bit) mode and four bytes per RGB LED in RGB (24-bit) mode.

Application Programming Interface

Application Programming Interface (API) routines allow you to control and configure the component using software. The following table lists and describes the interface to each function. The subsequent sections cover each function in more detail.

By default, PSoC Creator assigns the instance name "StripLights_1" to the first instance of a component in a given design. You can rename the instance to any unique value that follows the syntactic rules for identifiers. The instance name becomes the prefix of every global function name, variable, and constant symbol. For readability, the instance name used in the following table is "StripLights".

Function	Description
void StripLights_Start()	Configures and starts component.
void StripLights_Stop()	Disables interrupt.
void StripLights_DisplayClear()	Clears the screen with a given color.
void StripLights_MemClear()	Clear display memory, but don’t update display.
void StripLights_Trigger()	Update entire display with memory data.
void StripLights_Pixel()	Set the LED at a given (X,Y) location the specified color
Void StripLights_GetPixel()	Return the color of a specific pixel
void StripLights_DrawLine()	Draw line of specified color from (x1,y1) to (x2,y2)
void StripLights_DrawRect()	Draw rectangle from upper corner (x1,y1) to lower corner (x2,y2)
void StripLights_DrawCircle()	Draw circle from point (x,y) with specified radius and color.
void StripLights_ColorInc()	Return next color in color wheel, with specified increment value.
void StripLights_Dim()	Dim next triggered update by 1, ½, ¼, 1/8, or 1/16
void StripLights_Ready()	Check if display update is complete
void StripLights_SetFont()	Set font type (Not implemented at this time)
void StripLights_PutChar()	Place character at (x,y) with specified foreground and background color
void StripLights_PrintString()	Print string at (x,y) with specified foreground and background color
void StripLights_CIRQ_Enable()	Enable row interrupt (enabled by default)
void StripLights_CIRQ_Disable()	Disable row interrupt.
void StripLights_FIRQ_Enable()	Enable LED interrupt (enabled by default)
void StripLights_FIRQ_Disable()	Disable LED interrupt.

void StripLights_Start(void)

Description: Initialize hardware, clear display memory, and enable interrupts.

Parameters: None

Return Value: None

Side Effects: None

void StripLights_Stop(void)

Description: Disable StripLights component.

Parameters: None

Return Value: None

Side Effects: None

void StripLights_DisplayClear(uint32 color)

Description: Clears the display memory with the given color, then updates the LEDs.

Parameters: uint32 color: Color in which to clear the display.

Return Value: None

Side Effects: All display data will be overwritten by the given color.

void StripLights_Mem_Clear(uint32 color)

Description: Clears the display memory with the given color, but does not update the LEDs.

Parameters: uint32 color: Color in which to clear the display.

Return Value: None

Side Effects: All display data will be overwritten by the given color.

void StripLights_Trigger(void)

Description: Starts the transfer of the display RAM to the LEDs. Use the StripLights_Ready() function to determine when entire transfer is complete.

Parameters: None

Return Value: None

Side Effects: None

void StripLights_Pixel(uint32 x, uint32 y, uint32 color)

Description: Set the pixel at location (x,y) with given colot.

Parameters: uint32 x: Horizontal position of the LED to modify
uint32 y: Vertical position of the LED to modify
uint32 color: Color to set the LED.

Return Value: None

Side Effects: None

uint32 StripLights_GetPixel(uint32 x, uint32 y)

Description: Return the color of the pixel at location (x,y).

Parameters: uint32 x: Horizontal position of the LED to modify
uint32 y: Vertical position of the LED to modify

Return Value: uint32: Color of pixel

Side Effects: None

void StripLights_DrawLine(int32 x0, int32 y0, int32 x1, int32 y1, uint32 color)

Description: Draws a line in the graphics memory from (x0,y0) to (x1,y1) with specified color.

Parameters: None

Return Value: int32 x0, y0: First endpoint of the line to be drawn.
int32 x1, y1: Second endpoint of the line to be drawn.
uint32 color: Color to set the LED.

Side Effects: None

void StripLights_DrawRect(int32 x0, int32 y0, int32 x1, int32 y1, int32 fill, uint32 color)

Description: Draws a rectangle in the graphics memory from upper left corner (x0,y0) to lower right corner (x1,y1) with specified color and fill option.

Parameters: None

Return Value: int32 x0, y0: First endpoint of the line to be drawn.
int32 x1, y1: Second endpoint of the line to be drawn.
int32 fill: Fill rectangle if fill is non-zero.
uint32 color: Color to set the LED.

Side Effects: None

void StripLights_DrawCircle(int32 x0, int32 y0, int32 radius, uint32 color)

Description: Draws a circle in the graphics memory with center at (x0,y0) with the given radius and color.

Parameters: int32 x0, y0: Location of center of circle.
uint32 radius: Radius of circle.
uint32 color: Color of circle.

Return Value: None.

Side Effects: None

void StripLights_ Dim (uint32 dimLevel)

Description: This function sets the entire display to the given dim level..

Parameters: uint32: Level to dim the LED display.

- 0: No dimming
- 1: Dim to 50%
- 2: Dim to 25%
- 3: Dim to 12.5%
- 4: Dim to 6.3%

Return Value: None

Side Effects: None

DC and AC Electrical Characteristics

TBD..

Component Changes

This section lists the major changes in the component from the previous version.

Version	Description of Changes	Reason for Changes / Impact
1.3		Initial document
1.5	Added command descriptions	
2.2	Added support for WS2812	