Mayhew Labs

Rotary Encoder LED Ring
User Guide

Table Of Contents:

Introduction

Since rotary encoders have no start or end point, knobs without a position indicator are typically used – as opposed to potentiometers, where an indicator knob is suitable. Many applications require a visual representation of how a rotary encoder is reacting to user input as well as the control's current position. The Rotary Encoder LED Ring offers a solution with a ring of 16 LEDs that surround the encoder. The designer may implement any desired sequence on the LEDs by communicating with the onboard series-to-parallel constant current shift register.

Hardware Description

The Rotary Encoder LED Ring has two separate sections: a rotary encoder breakout and a collection of LEDs controlled by a shift register. A microcontroller can be used to obtain encoder data and set LED data.

A rotary encoder may be mounted to the printed circuit board in two separate ways. The PCB has (1) mounting-tab holes and soldering positions for a rotary encoder as well as (2) a hole for the encoder shaft: the encoder may be soldered to the PCB or fitted through the shaft hole. Compatible rotary encoders are described below in the '**Compatible Hardware**' section.

The PCB has 15 LEDs in a 270˚ arc and one LED at the bottom of the arc; they are interfaced with a Texas Instruments TLC5925 16-bit shift register. The interface uses a standard Serial-Data/Clock/Latch-Enable 3 wire connection that is further described with timing diagram in the TLC5925 datasheet: http://focus.ti.com/lit/ds/symlink/tlc5925.pdf.

A thru-hole resistor can be used to increase the brightness of the LEDs and is described further in the '**Adjusting LED Current and Brightness**' section.

Please see the '**Pin Descriptions**' section for more information about the 10 available connections.

Compatible Hardware

The Rotary Encoder LED Ring is compatible with any microcontroller with at least 2 input lines (for the rotary encoder) and 3 output lines (for the serial interface) available.

It can accept knobs up to 3/4-inch diameter without covering the LEDs. Image 1 shows how a 3/4" knob appears on a shaft-mounted encoder.

Image 1: Rotary Encoder LED Ring with a 3/4-inch knob

The Rotary Encoder LED Ring is compatible with many rotary encoders; encoders that have either a compatible footprint or shaft diameter (7mm or less) may be used. The PCB is capable of encoders with or without a push switch. Table 1 lists a small set of compatible thru-hole encoders and Figure 1 shows the PCB footprint to consult for encoders not listed.

Table 1: Partial List of Compatible Devices

Brand	Models
Alpha	318ENC Series
Alps	EC11, EC12
Bourns	PEC11, PEC12, PEL12, PES12, PEC16
CTS	290 Series
Panasonic	EVE Series
Sparkfun	SKU: COM-09117

Figure 1: PCB Footprint for Compatible Rotary Encoders

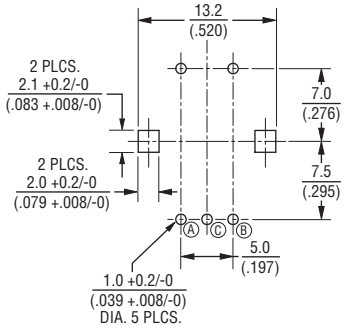

Image Credit: Bourns PEC11 Datasheet

Pin Descriptions

Pin	Description	Section
GND	Ground, zero voltage reference	Power
VCC	3 to 5.5 Volts (Yellow, Green, Red LED models) ** **BLUE LED model requires 3.6 to 5.5 Volts** **	
ENCA	Rotary Encoder A Terminal	Rotary Encoder
ENCB	Rotary Encoder B Terminal	
SWTCH	Rotary Encoder Switch	
SDI	Serial Data Input	TLC5925
CLK	Clock	
LE	Latch Enable	
OE (Active Low)	Output Enable; tie to ground for constant operation	
SDO	Serial Data Output – for daisy chaining units	

Example Hardware Connection for Arduino:
The following is an example of how the Rotary Encoder LED Ring might be connected to an Arduino. The example code listed below follows this set of connections.

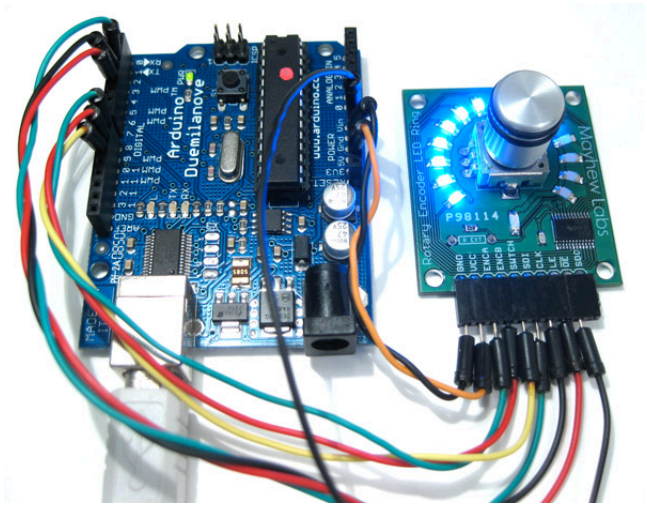

Image 2: Connections to an Arduino

Rotary Encoder LED Ring Pin	Arduino Pin
GND	GND
VCC	5V
ENCA	8 (Digital)
ENCB	9 (Digital)
SWTCH	10 (Digital)
SDI	2 (Digital)
CLK	3 (Digital)
LE	4 (Digital)
OE (Active Low)	GND
SDO	Not connected

Table 2: List of pin connections to an Arduino

Example Code for Arduino:
Three examples of how the Rotary Encoder LED Ring might be used can be found in the Arduino example code available at: www.mayhewlabs.com/products/rotary-encoder-led-ring

If an Arduino is not the desired target device, the example code gives enough detail to outline the procedures necessary to implement an interface on other programmable devices.

Adjusting LED Current and Brightness:

The Rotary Encoder LED Ring ships with dim LEDs so the end user may increase the brightness to a level appropriate for the application. The current to each of the 16 LEDs is constant and is set per the TLC5925 datasheet. A surface mount 22kΩ resistor is provided and a parallel thru-hole resistor may be added to increase the brightness by lowering the resistance. Please reference Equation 1 and Figure 2 to set the current output to the LEDs. Using a parallel resistance calculator like http://www.sengpielaudio.com/calculator-paralresist.htm or Equation 2 can help determine what resistor value to use for the thru-hole resistor. With the surface mount 22kΩ resistor, the output current is set to about 1mA. Adding a 1kΩ resistor thru-hole will increase the LED brightness to their highest setting.

A potentiometer or voltage controlled resistor may also be used to adjust the brightness on the fly. Care must be taken to not increase the current beyond the capability of the LEDs (20mA to 30mA max).

Equation 1: Current Output to Each LED

$$I_{OUT} = \frac{1.21V}{R_{ext}} \times 18$$

Figure 2: Relationship between I_{OUT} and R_{EXT}

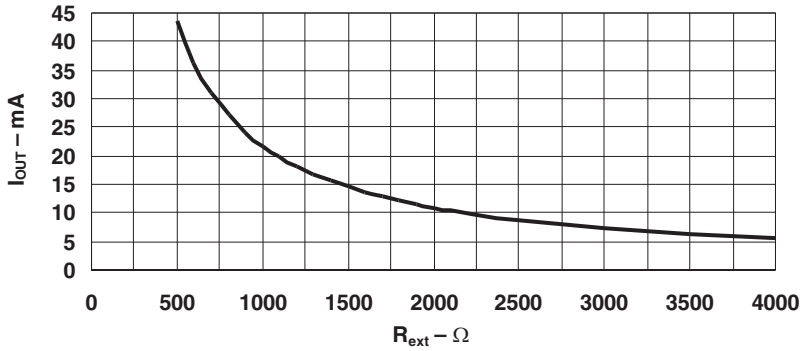

Image Credit: Texas Instruments TLC5925 Datasheet

Equation 2: Parallel Resistance Total (Surface Mount and Thru-Hole)

$$R_{ext} = \frac{R_1 \times R_2}{R_1 + R_2}$$

Diffusing the LEDs:
Some applications may require LEDs that have more uniform lighting or appear more as an indicator than a light source. Sanding the clear lens of the LEDs, roughening it up to create a diffused lens can accomplish this. 400 grit sand paper or similar works well.

Setting the LED Sequence:
There are primarily two methods to create an LED sequence to be outputted to the shift register. The first, and easiest, is to create a sequence of 'images' that will be stepped through as the encoder is rotated. The second is to use bitwise operations (AND, OR, XOR) to build and modify the 'image' that is currently on the LEDs. This method is how the example code turns on the bottom LED while not changing the current image otherwise.

As an example, the image to output that will turn on one additional LED each step looks like:
0000000000000001 (hex 0x01)
0000000000000011 (hex 0x03)
0000000000000111 (hex 0x07)
and so on. This sequence can be written directly to the shift register.

If the same output was created using XOR or OR, the sequence would look like:
0000000000000001 (hex 0x01)
0000000000000010 (hex 0x02)
0000000000000100 (hex 0x04)
and so on. An image would be OR'ed with the previous image and the result would be the same as the first example.

For details about using bitwise operations, please see the Arduino Reference:
http://arduino.cc/en/Reference/BitwiseAnd

Using the map() function, it is possible to scale the speed of the LED output sequence relative to the encoder rotation. For example, when rotating the rotary encoder one full revolution, one additional LED is turned on. See the Arduino Reference: http://www.arduino.cc/en/Reference/Map