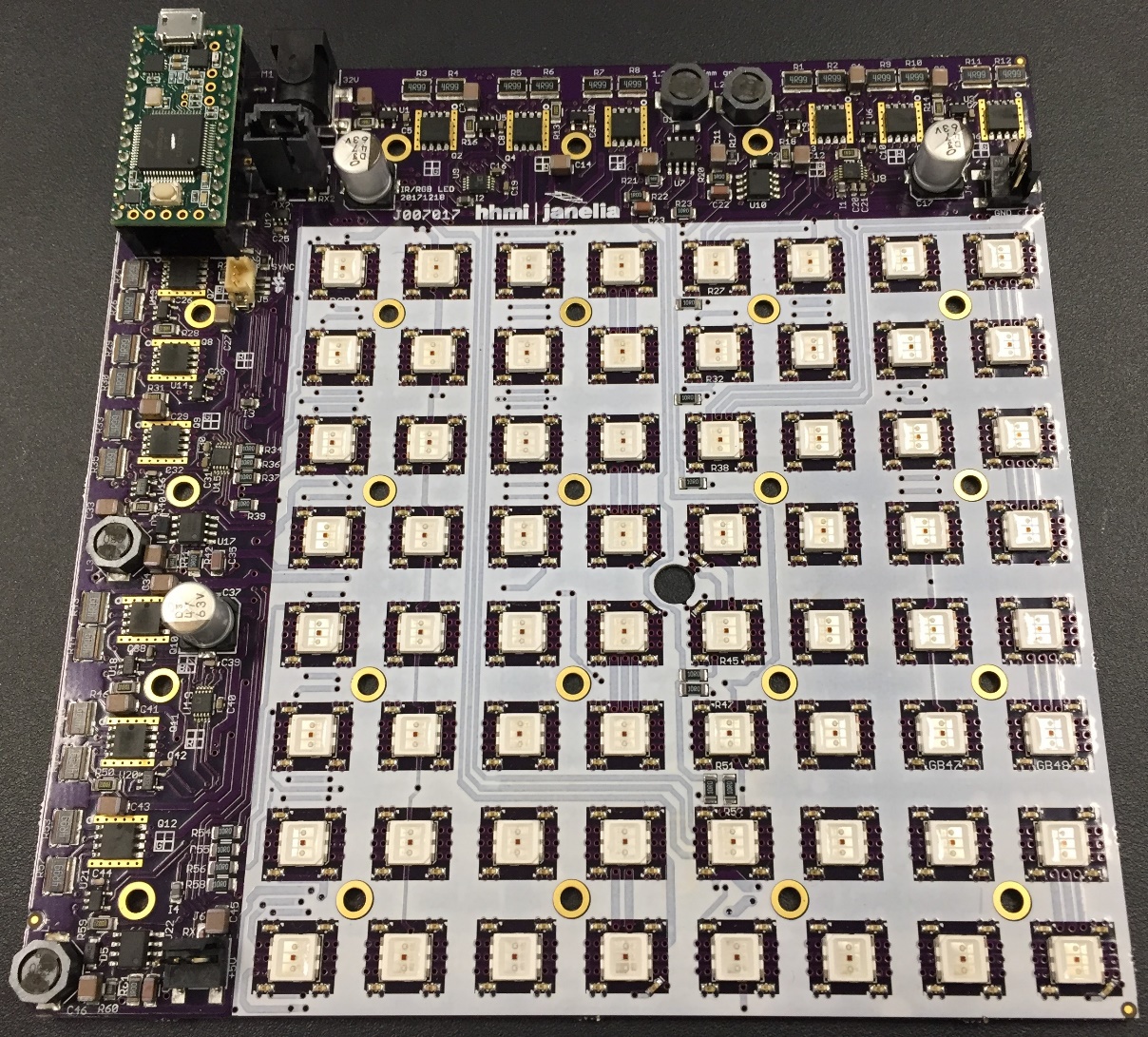
RGB-IR LED Panel

J00717 Version REV A

****

Steve Sawtelle  
Lead Electrical Engineer, ID&F  
(571) 209-4158

Contents

[System Overview 2](#_Toc516219834)

[Hardware Development 2](#_Toc516219835)

[Connectors 3](#_Toc516219836)

[Setup and Deployment 4](#_Toc516219837)

[Command Set 5](#_Toc516219838)

[Intraboard Commands 7](#_Toc516219839)

# System Overview

The RGB-IR LED Panel is used to provide red, green, blue, and/or IR lights of various intensities. A built in micro-processor controls the panel via a USB-Serial interface. This allows the panel to directly control pulsing of the LEDs, offloading the task from the host computer, and ensuring accurate timing. The panel is divided into four quadrants that can be independently set up with different colors and intensities. The lit area of the panel is 4.8” x 4.8” (122mm x 122mm). The overall size of the panel is 6” x 6” (152.6mm x 152.6mm). Four panels can be combined to form a lit area of 9.6” x 9.6” (244mmx 244mm), all controlled through one USB connection. A 3-pin connector can be used to drive an indicator LED and/or perform sync in and out functions. Two TTL serial connections allow the panel to be communicate with other devices.

# Hardware Development

The board incorporates a Teensy 3.2 processor with code developed under the Teensyduino environment to make code updates and changes easy. The RGB LEDs are controlled via linear drivers to provide constant levels with no PWM artifacts. The linear control driver is through a standard FET with a low-side current sense resistor driving the feedback control via an op-amp. Since there are two series strings of LEDs in each quadrant, a dual, matched FET is used to passively regulate the second string. Using linear regulators does increase power dissipation, which should be low if the LEDs are pulsed at low duty cycles. The IR LEDs are PWM controlled to lower heat dissipation as they are typically always on. The IR PWM artifacts are also out of the detection range of most test animals. Multiple panels are controlled by a serial communications link that is daisy-chained from board to board. The RGB LEDs are on 0.6” (15.24mm) grid and the IR LEDs are on a 0.3” (7.62mm) grid.

*Schematic*

See project file

*Printed Circuit Board*

See project file

*Materials*

See project file

# Connectors

**J1** – power barrel 2.5 x 5.5 mm

Power In – 28-32VDC @ 5 Amp minimum; 28 VDC recommended

**Mini-USB** on Teensy-3.2

Connects to host computer for control and firmware updates

**J5** – 3 pin latching, Hirose DF13 series

Marker LED connection

1. Teensy pin 20 through 332 ohms, intended to drive a marker LED
2. Teensy pin 19 through 100 ohms, intended for a sync input
3. ground

**J2-J3** – 3 pin connector; 0.1 center

Board daisy chain

1. ground
2. TX/RX
3. +5VDC

# 

# Setup and Deployment

LED Intensity

See the spreadsheet file: led current\_voltage\_intensity\_28V.xlsx

Power Supply

A power supply with a rating of 28 to 36 volts is required. The lower voltage will produce less heat dissipation and is highly recommended for high or long illumination. The current required depends on which LEDs are illuminated and at what maximum intensity. The BOM has an area at the bottom to enter the anticipated illumination (in %) for each color and it will show the total current requirement. Multiple board can be run from a single power supply but it must supply the total current needed by all boards. A Mean Well ENP-360-24 can supply up to 13 amps at 28 Volts (voltage setting needs to be adjusted up from factory setting of 27.6V). Screw connections allow several DC power cables to be attached, such as MPD 172-4208. The Cincon TRH100A280-12E13-Level-VI power supply can supply 28V at up to 3.54 volts and can be plugged directly into the RGB board. A single board, with a power requirement of less than 1.2 amps, could use an XP Power ACM36US30, which also can plug directly into the board.

Mounting and heatsinking

The panel needs to be mounted on a 6” x 6” (152.4mm x 152.4mm) non-conductive heat sink using a 5x5 array of 4-40 (or 3mm) tapped holes on 1.2” (30.48mm) centers. A diffuser positioned at least 0.5” (12.7mm) above the panel will provide uniform IR lighting over the lit area. The panel runs off a single 28-volt power supply, although up to 36 volts can be used if duty cycles are low or good heat sinking is employed. At lower duty-cycles the major heat source is the linear regulators (12 large flat ICs), especially the red drivers. If continuous RGB illumination is used then the LEDs become the major heat source. In general, at low illumination and/or duty cycles, a flat plate at least ¼” (6mm) can be used. Alternatively, a finned heat sink such as Aavid 601403B06000 can be used. At higher power levels and longer on times, liquid cooling will be needed using a cold plate such as Aavid 416101U00000G and a cooler such as a Koolance EXT-440CU computer cooler. A non-conductive thermal pad is recommended between the board and the heatsink, such as Berqquist GPHC5.0-0.020-02-0816.

Firmware Development

The firmware is written under the Arduino programming environment with Teensy extensions added in. There are three compiler defines that affect the operation of the firmware:

USE\_SCALING - normally undefined

If undefined, the brightness of the optogenetic LED is set in a linear fashion. If defined the brightness is set with pseudo-log scaling, such that up to 40%, the LED brightness increases in a small step linear fashion, and above 40%, it increases logarithmically.

LINEAR\_DRIVER – normally defined

This should be defined when used with the current panel design. Earlier panels used a PWM brightness control for the optogenetic LEDs and required an offset in the brightness setting.

USE\_EXT\_TRIG - normally undefined

If defined, an external trigger can be used to turn the optogenetic LEDs on and off.

# Command Set

The host communications is via a USB-serial link. The Teensy serial driver must be loaded on the host (available from PJRC.com). The link is baud-independent. The communications protocol is based on an ASCII commands with various parameters. Commands are not case sensitive and must be terminated with a carriage return character (‘\r’, 0x0D). Basic error checking is done and ‘cmderr n’ is returned with ‘n’ being a negative integer that can be used to determine the type of error. Commands may return other information to the Host to indicate status. These are still under development.

The following commands are supported:

**RESET** – reset all boards to their power-on state. This should always be sent at the beginning of a session or after power has been cycled.

**RED p** – set the red LEDs to pc % brightness

**BLUE p** (or)

**BLU p** – set the blue LEDs to pc % brightness

**GREEN p** (or)

**GRN p**– set the green LEDs to pc % brightness

These commands control the LED intensity for red, blue, and green respectively. The percentage intensity is between 0 and 100% and can be a floating point value. Low values (<5%) may show different intensities across the LEDs since they are close to their turn-off state. NOTE: This command only sets intensity; it does not turn the LED on. Use On and OFF to control LED state.

**IR p** – set IR level to ‘p’ percent of full brightness.

A setting below about 15% will turn off the IR LEDs. There is no separate TTL control. Values near 15% may have inconstant individual LED brightness. It is best to set brightness to 0% if the LEDs should be off. Use values above 15% to turn on the LEDs to a percent of full brightness.

**ON p q** – turn on RGB LEDs to their last setting, optional panel(s) and quadrant(s)

**OFF p q –** turn off RGB LEDs, optional panel(s) and quadrant(s)

When the RGB LEDs are set to an intensity, that value is stored. Use ON and OFF to turn RGB LEDs on and off.

p = panel(s): 0 = all, or 1, 2, 3, or 4 for an individual panel

q = quadrant(s): 0 = all, or 1, 2, 3, 4 for an individual quadrant

**PULSE width, period, number, off, wait, iterations,** **color** - pulse setup command individual color

The PULSE command sets up a pulse train to light Optogenetic LEDs that have been set up with an ON or PATT command. The pulse has: on delay, pulse width, period, number of cycles, off delay, wait time, and number of iterations.

color: Which color the pulse command is for (‘R’, ‘G’, and/or ‘B’) – the three characters can be used in any combination. If different pulse setups are required for different colors, then use separate PULSE commands

width: The pulse width (on time) in milliseconds from 1 to 30000

period: The period of the pulse (on time plus off time) in milliseconds from 1 to 30000. If the width and period are equal, then the LEDs are on constantly during the pulse train.

number: The number of pulses in a pulse train. One pulse train starts with an on time and ends ‘number’ of pulse later after the last off time.

off: The off time after the pulse train completes in milliseconds from 0 to 30000. This provides a means to have a ‘dead time‘ between pulse trains if iterations are used. Note that the total off time from when the last pulse on time ends to the next pulse on time starts is equal to: off + period – width.

wait: Delay the start of the pulse sequence in seconds from 0 to 120. This delays the start of the very first pulse after the RUN command is sent. It is not repeated again until RUN is resent. A floating point value can be used here to set timing to the millisecond level.

iterations: The number of time the pulse train is repeated from 0 to 30000. A ‘0’ value runs the pulse train continuously until a STOP command is sent.

color: which color(s) to turn on and off with this pattern

**RUN** c- Start running the pulse sequence, optional color(s)

* c – color(s) any combination of ‘R’, ‘G’, ‘B’

**STOP** c- Stop running the pulse sequence and reset to the beginning, optional color(s)

* c – color(s) any combination of ‘R’, ‘G’, ‘B’

**???** - Returns the version number and other parameters

# Intraboard Commands

This section describes the commands sent between boards. The current RGB panel with serial communications between panels (J007017) is based on a TTL serial link using Serial1. Each panel has an input and output line. All boards have a Teensy installed, but only one (main panel) is connected to the host. The host sends all commands to all or any other panel(s) via the main panel. The main panel interprets the command, and if is destined for one or more other panels, it sends the command on via Serial1. The incoming serial connection to the panel is always connected to the micro’s Serial1 receive port and the outgoing serial connection leaving a panel is routed through a TX/RX switch to route signals. The panel’s output signal can be switched between a direct connection to the input (Parallel Mode) or connected to the panel’s Serial1 transmit port (Series Mode).

Initially, all panels boot up in Parallel Mode. On reset, the main panel will send a Series command to put the attached panels in daisy-chain mode. This is necessary so that the boards can enumerate so they know what position they are in. An initial Enumerate1 command by the main panel propagates to the second panel, which configure itself as panel 2, and it the sends out an Enumerate2 to the second panel. This panel then configures itself as panel 3, and sends an Enumerate3 command to the last panel. After allowing time for the commands to propagate, the main panel sends a Parallel command to set the panels back to normal operation. Then any commands from the main panel is passed directly to all panel simultaneously to minimize delay of commands between panels.

Commands

00pp bbbb Turn off selected quadrants on selected panels

pp = panel (00 = all, 01 = 1, 10 = 2, 11 = 3)

bbbb = quadrant bits quadrant bits(0b0001 is upper left, 0b0010 upper right, 0b1000 lower right, 0b0100 lower left

01pp bbbb Turn on selected quadrants on selected panels

pp = panel (00 = all, 01 = 1, 10 = 2, 11 = 3)

bbbb = quadrant bits quadrant bits(0b0001 is upper left, 0b0010 upper right, 0b1000 lower right, 0b0100 lower left

10pp qqcc Set intensity command of selected panel(s), quadrants(s) , and color(s) to last Intensity value (0-4095)

pp = panel (00 = all, 01 = 1, 10 = 2, 11 = 3)

qq = quadrant number 00 is upper left, etc

cc = color (00 = BLU, 01 = RED, 10 = GRN, 11 = IR)

1100 iiii Set the lower 4 bits of the intensity value

iiii = intensity value nibble

1101 iiii Set the middle 4 bits of the intensity value

iiii = intensity value nibble

1110 iiii Set the highest 4 bits of the intensity value

iiii = intensity value nibble

1111 00xx Enumerate

xx = 01 sent by main board, 10 sent by next panel, 11 sent by third panel

1111 0100 Set RX/TX switch to Series mode

1111 0101 Set RX/TX switch to Parallel mode

1111 1ddd Turn color DACs on or off on all panels (used for fast response, especially for pulsing

ddd = color DACs: bit 2 = red, bit 1 = green, bit 0 = blue