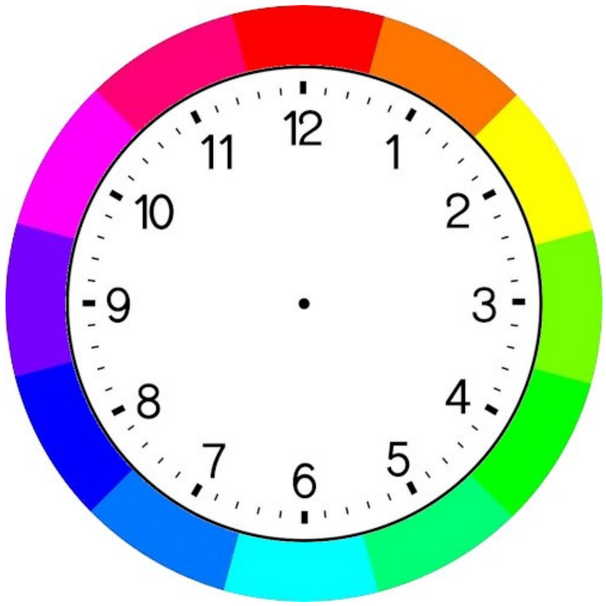
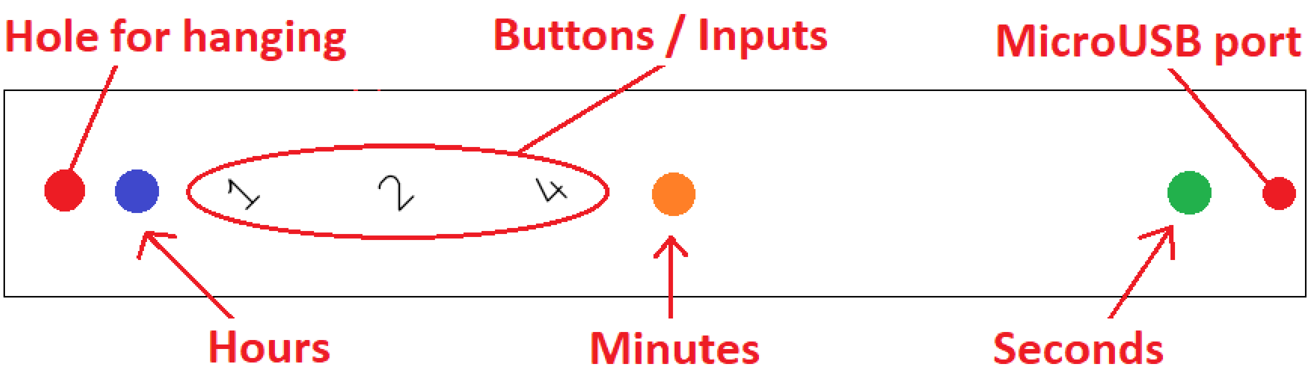
The Spectacular Clock

Created, Programmed, & Designed by: Ryan Jardina



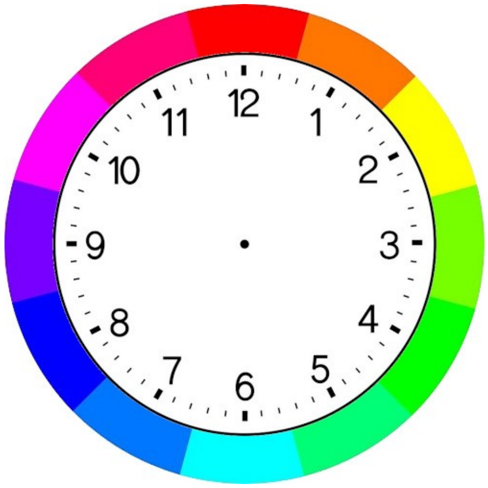
Below is a diagram of The Spectacular Clock. The Pixels change color with time.



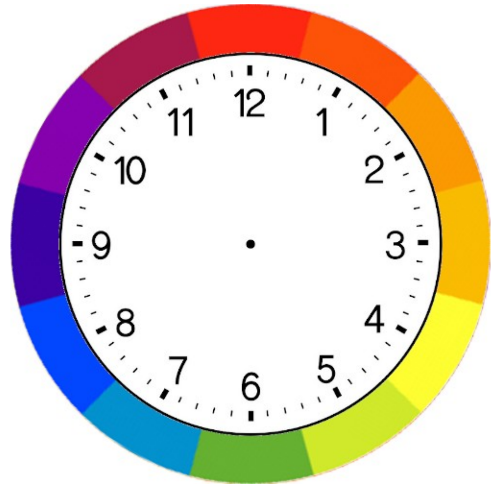
Time on the diagram is 8:05:20, based on the color wheel above it.

Inputs numbers are as follow:
1 = Add one hour to current time
2 = Add one minute to current time
3 = Adjust the brightness of the clock (3 levels)
4 = Add 3 seconds to current time
5 = Switches Hours' & Seconds' Pixels
6 = Shift seconds' color to 12 / undo (see page 3)
7 = Resets clock & goes to initial setup. (see page 2)
(3, 5, 6, & 7 are combinations simultaneously add/push 1, 2, & 4 for these inputs)

Initial Setup

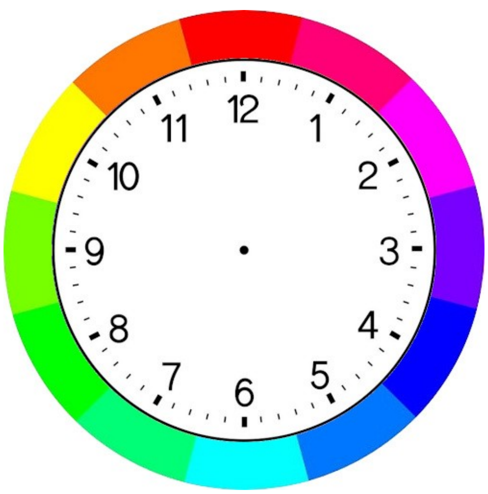


RGB color wheel
(Red, Green, Blue)

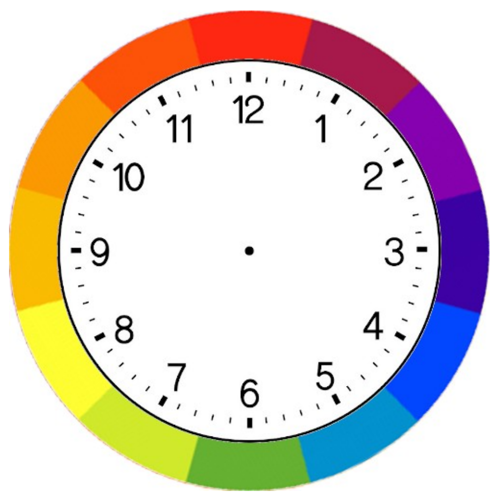


RYB color wheel
(Red, Yellow, Blue)

To Change order press “ 2 ”



RBG color wheel
(Red, Blue, Green)



RBY color wheel
(Red, Blue, Yellow)

To choose RGB or RBG press “ 1 “
To choose RYB or RBY press “ 4 “

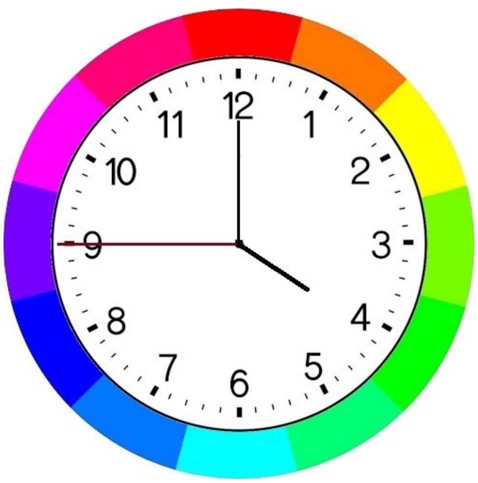
During setup the Hours’ pixel shows the RGB/RBG color wheel & the Seconds’ pixel shows the RYB/RBY color wheel. Minutes’ pixel, if dim to bright shows R?B, if bright to dim shows RB?.

Explaining Input 6

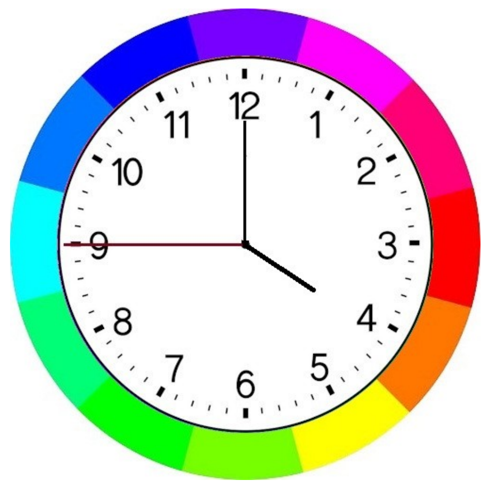
After the initial setup all the pixels will be Red & the seconds will began to tick.

Say you choose RGB. The minutes will not tick until the seconds reached Red again. The purpose of Input 6 is not being stuck with Red at the top of the clock representing 12 in hours or 00 in minutes & seconds.

For example you want purple at the top of the clock. So you either add seconds or wait for the seconds to turn purple. Which is at the 45 second mark. After the clock's seconds turn to the color you want, press buttons 2 + 4 (Input = 6). If the clock was at 4:00:45 (Green, Red, Purple) after Inputting 6 the colors will now be Orange, Purple, Cyan (see figure below).



Before applying input 6



After applying input 6

The time does not change (seconds will continue to tick). This option rotates the color wheel. If input 6 is apply to the clock again it will undo & revert back making Red being at the top of the clock again.

After the clock has been setup & power is reapplied there is a 10 second startup sequence. That begins & ends at top of the clock & showing color order.

Reason/History

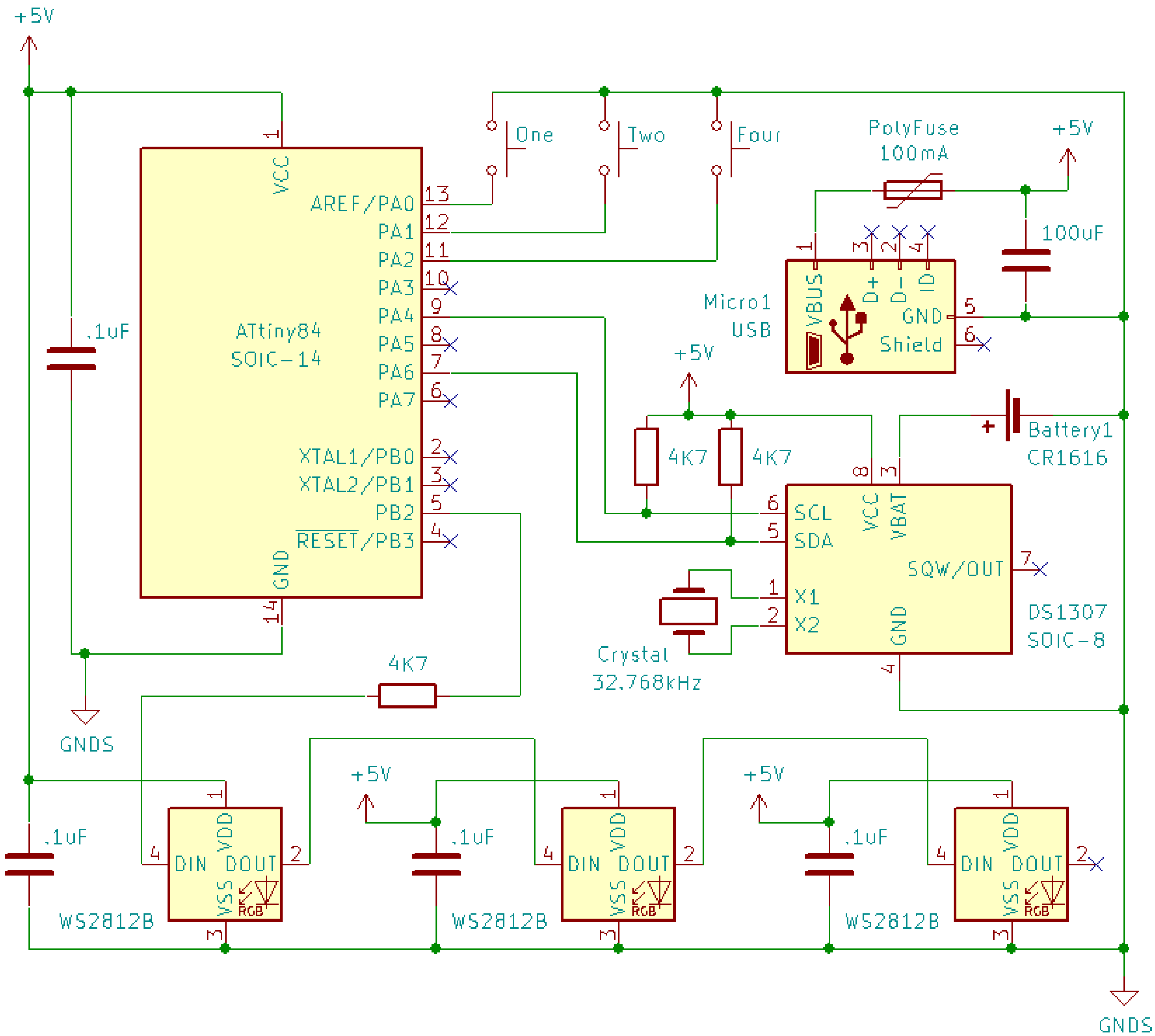
The initial reason for this clock is I’m not able to read a normal clock from across the room without eye glasses, but am able to see the three pixels & their color.

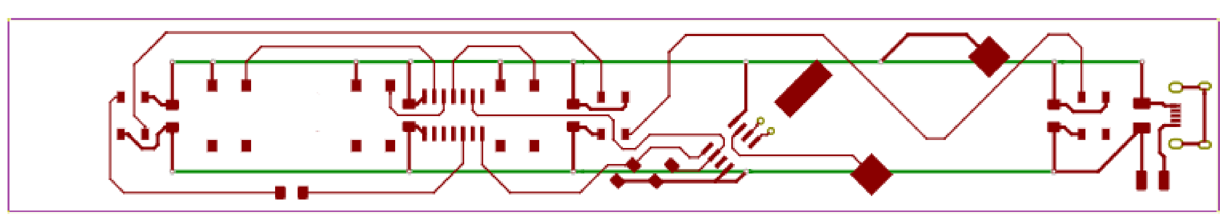
Theory of Operation

There are a total of 5 integrated circuits; Main chip (ATtiny84), Real Time Clock (RTC/DS1307Z), & 3 Addressable pixels (WS2812B). WS2812B have 3 LEDs built in; Red, Green, & Blue.

Each pixel require 24 bits to determine their color; 8 bits for Red, 8 for Green, & 8 for Blue. Using 8 bits gives 255 different intensity per color. By mixing these colors with their intensity any color on the color wheel can be produce.

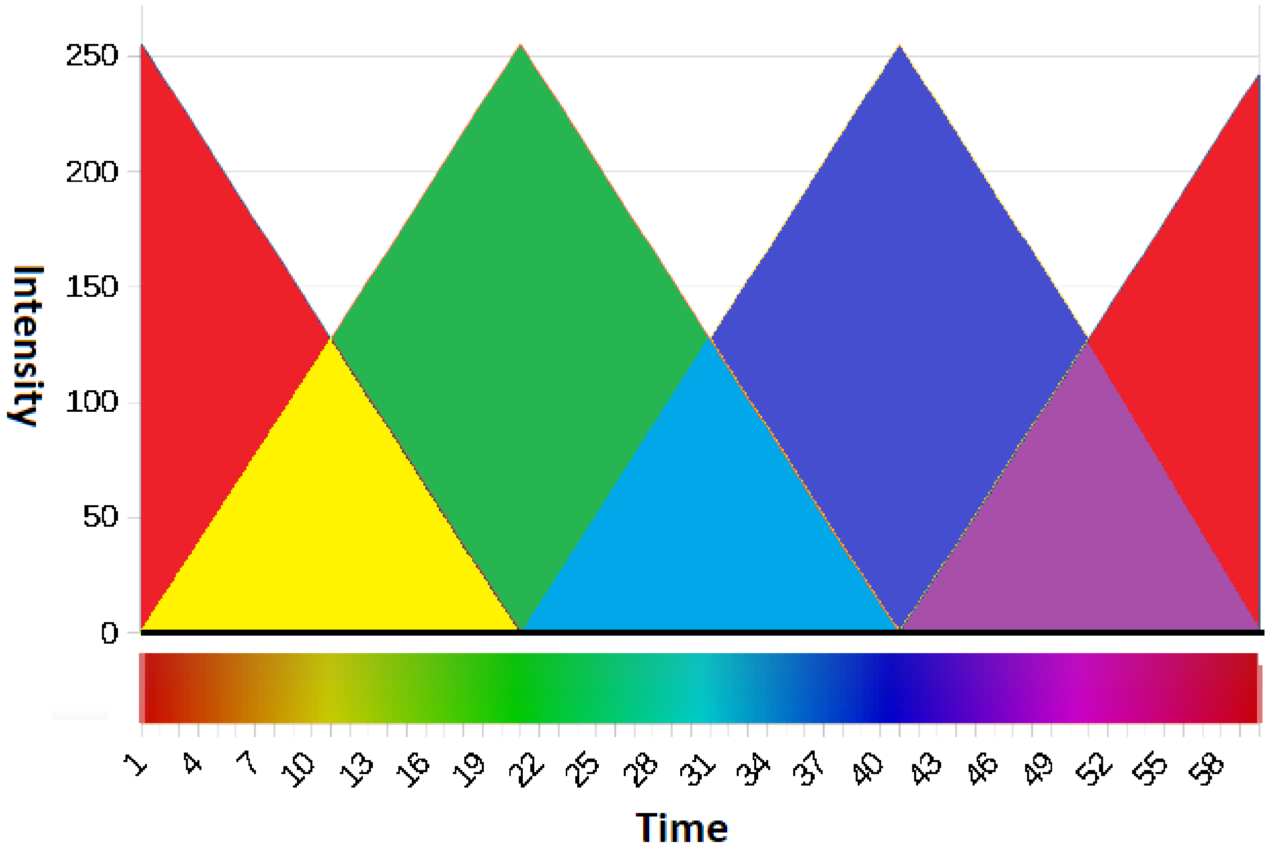
Main chip receives the time from the RTC. Runs the hours, minutes & seconds through mathematical equations. Which is similar (but not really) to three phase AC power. But instead of using sine waves it use triangle waves. The peak are 255 & any number below zero are rounded to zero.


+5V
ATtiny84
SOIC-14
VCC
GND
AREF/PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
XTAL1/PB0
XTAL2/PB1
PB2
RESET/PB3
.1uF
GNDS
One
Two
Four
PolyFuse
100mA
100uF
Micro1
USB
VBUS
D+
D-
ID
GND
Shield
4K7
Battery1
CR1616
DS1307
SOIC-8
SCL
SDA
X1
X2
VCC
VBAT
SQW/OUT
Crystal
32.768kHz
WS2812B
DIN
DOUT
VDD
VSS
RGB



Pixel Output

The pixels have 3 LEDs built in; Red, Green & Blue (RGB). Below is a graph over time of the intensity of the individual color LEDs, making up the RGB color wheel. If any single color is at it's peak it is 255. A blend of any two colors has a sum of 255.

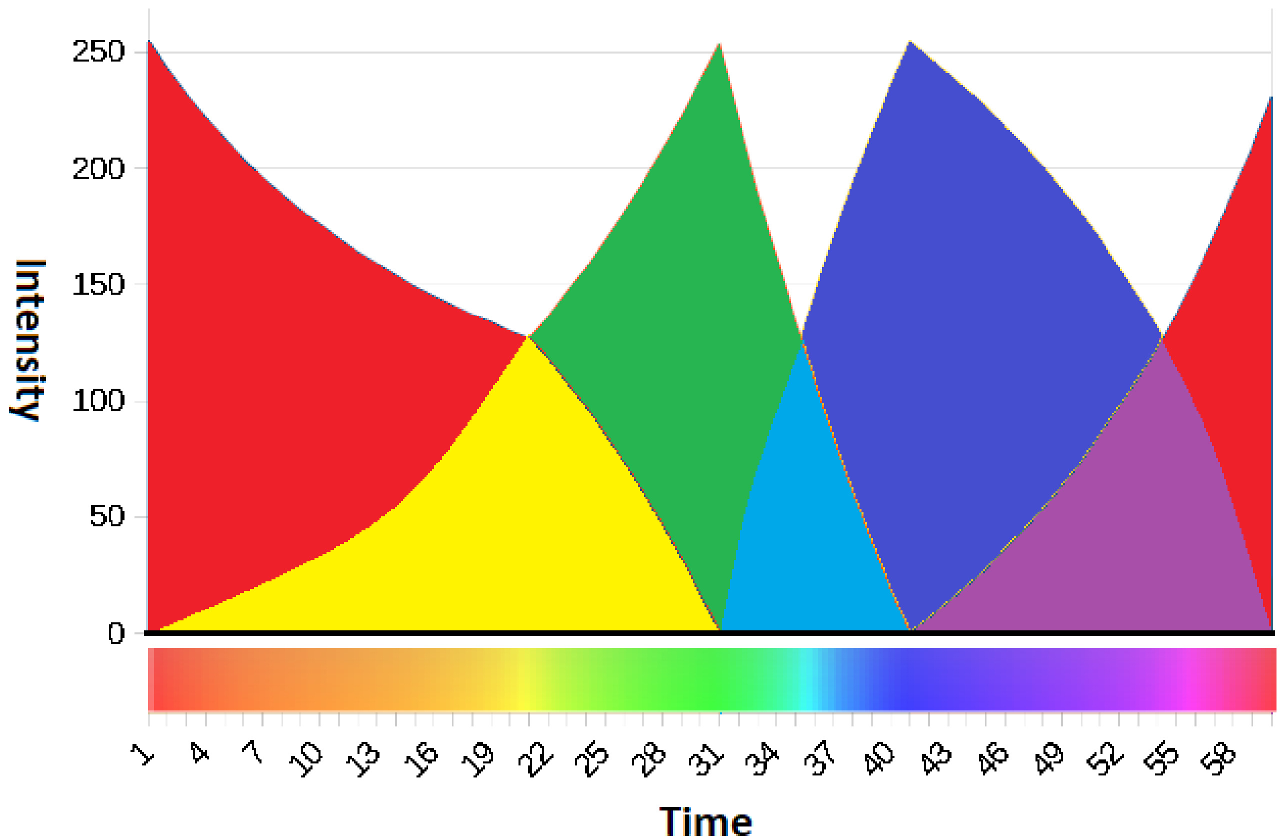



```
if (Direction == 0) {Angle[1] = (1.3333); Angle[2] = (.6666);}
   else {Angle[1] = (.6666); Angle[2] = (1.3333);}
 for (Q = 0; Q <= 2; ++Q) {
   X = Digit[Q] + Shift;
   for (Phase = 0; Phase <= 2; ++Phase) {
     RGB[Phase] = 63.5 + (2 * 192.2 / Pi) * asin(cos(2 * (Pi / 60) * X + Angle[Phase] * Pi)); }
     RGB[Phase] = max(RGB[Phase], 0); }
```

The math above is used to calculate the output of RGB.

RYB Output

The graph below is of the RYB color wheel output. Since the pixels only has RGB. The intensity over time of the RGB has to be distorted to output the RYB color wheel.




```
X = min(RGB[1], RGB[2]);
RYB[0] = RGB[0] + RGB[1] - X;
RYB[1] = RGB[1] + (2 * X);
RYB[2] = 2 * (RGB[2] – X);
I = RYB[0] + RYB[1] + RYB[2];
I = 255 / I;
RGB[0] = I * RYB[0];
RGB[1] = I * RYB[1];
RGB[2] = I * RYB[2];
```

<<== Stolen from
Paper: RYB Color Compositing
By: Junichi Sugita
& Tokiichiro Takahashi

Above is the second part of the calculations for RYB. The first part is actually the same from the RGB output. The RGB equations determined RYB instead of RGB. This math twist it to the output for the pixels.

Notes

The colors in this manual don’t represent 100% the output of the clock.

By holding down any button while powering on the clock after the initial setup, will make the seconds blink for the last three minutes of the hour.

If the receiver of this clock doesn’t not want it, regifting is totally fine. Use it as a bookmark or blow it up, or something else. I demand someone get at least a small amount of enjoyment from it.

The clear plastic that is rapped around the clock is suppose to be there. It keeps the board and the wood veneer together. The veneer is there to diffuse the light.

For some odd reason if you don’t have an old microUSB cable & wall changer to power the clock, The ones at a dollar store will work just fine.