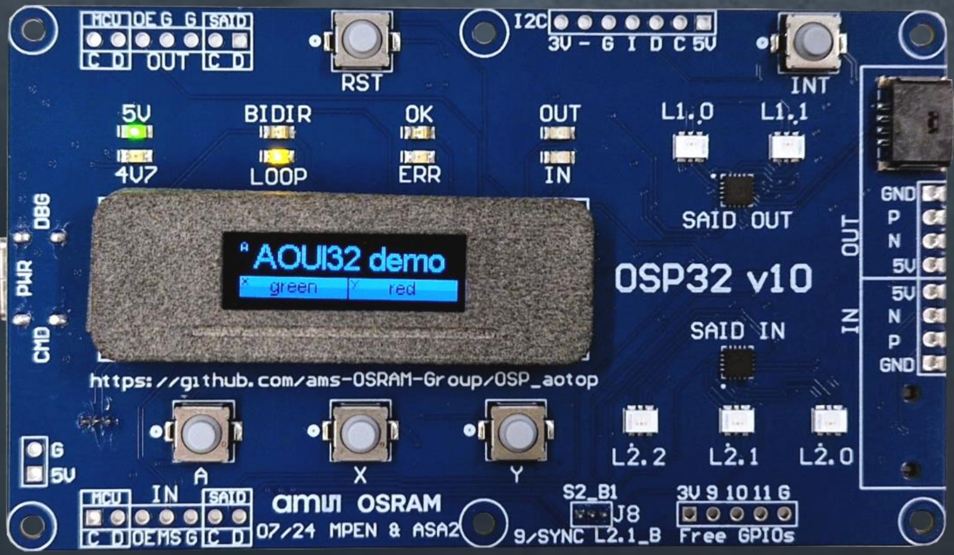
OSP on Arduino – Training – Appendix 2

Using the *Arduino OSP evaluation kit* – Uniform colors



2025 May 19

Uniform colors

Background

One of the selling points of OSP nodes of ams OSRAM is the ability to

(1) have two adjacent nodes show the same (specified) color
(2) keep that color when temperature changes

Applications that utilize these abilities can be called color-accurate or having uniform colors.

“*The same color point across the chain irrespective of ambient conditions*”

Some ams-OSRAM components like OSIRE E5515 or E3323 have a DMC (dot matrix code). This is key into a table of color calibration values.

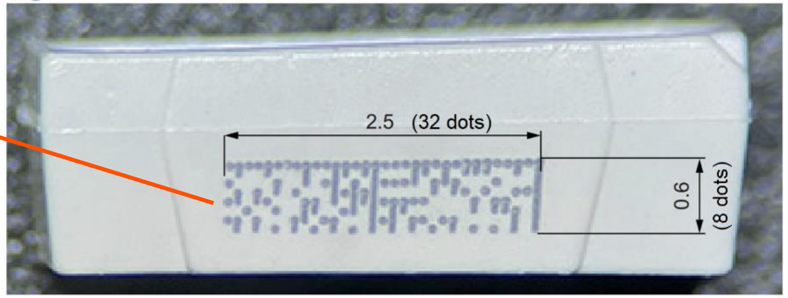


Figure 2: OSIRE® E5515 DMC

Calibration

Per node

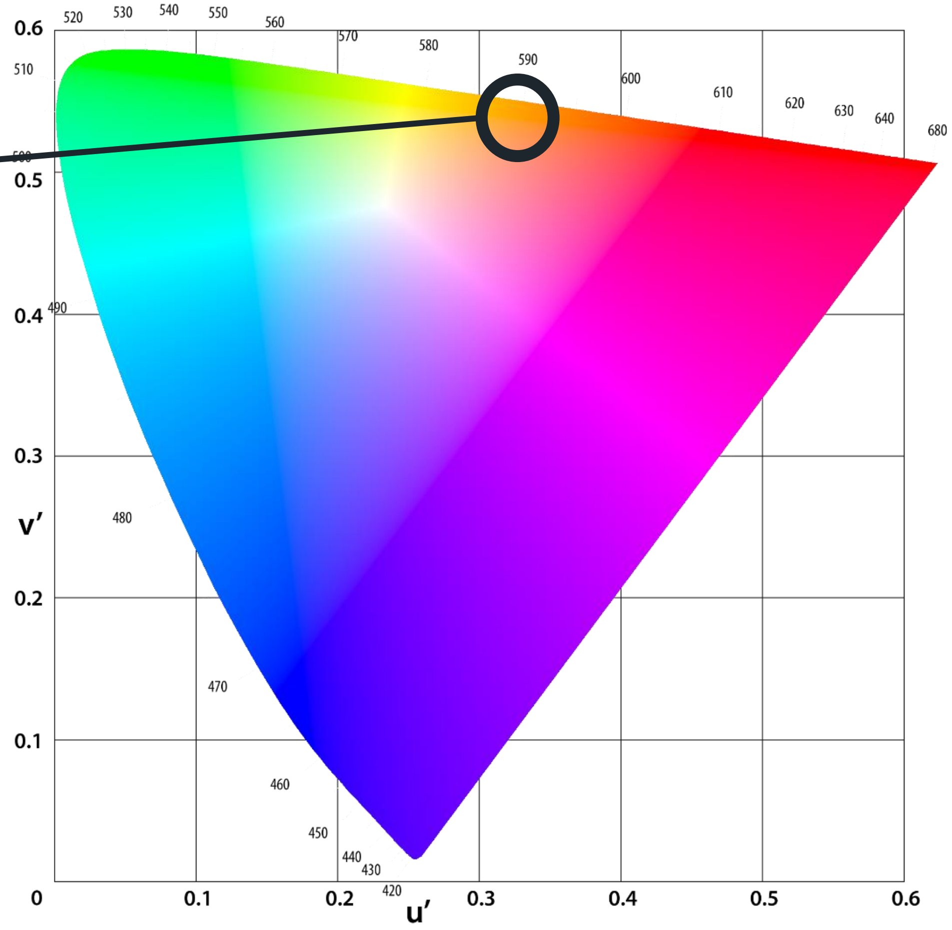
- To characterize a triplet (RGB module) the CIELUV model is used
- See https://en.wikipedia.org/wiki/CIELUV

- For example, OSRAM orange is approximately at u'=0.35 and v'=0.55
- This is just the color, luminosity (I_v) is the z-axis (not shown in triangle)

- Taking RGBI as example, calibration data for the three LEDs is stored in OTP:
  - 8 bits for u', 8 bit for v', and 12 bit for I_v (3.5 bytes in total)
  - Twice: once for night mode (10mA), once for day mode (50mA)
  - One LED thus takes 7 bytes, an RGB triplet (3 LEDs) takes 21 bytes

- Taking example `aomw_colordemo` as example
  - it uses a different color space: CIE 1931 (C_x, C_y, I_v)
  - see https://en.wikipedia.org/wiki/CIE_1931_color_space

- Bottom line
  The calibration data describes exactly the color of each of the three LEDs (red, green, and blue) of the triplet



am OSRAM

Computing PWM to reach target color

Color mapping

Graphically

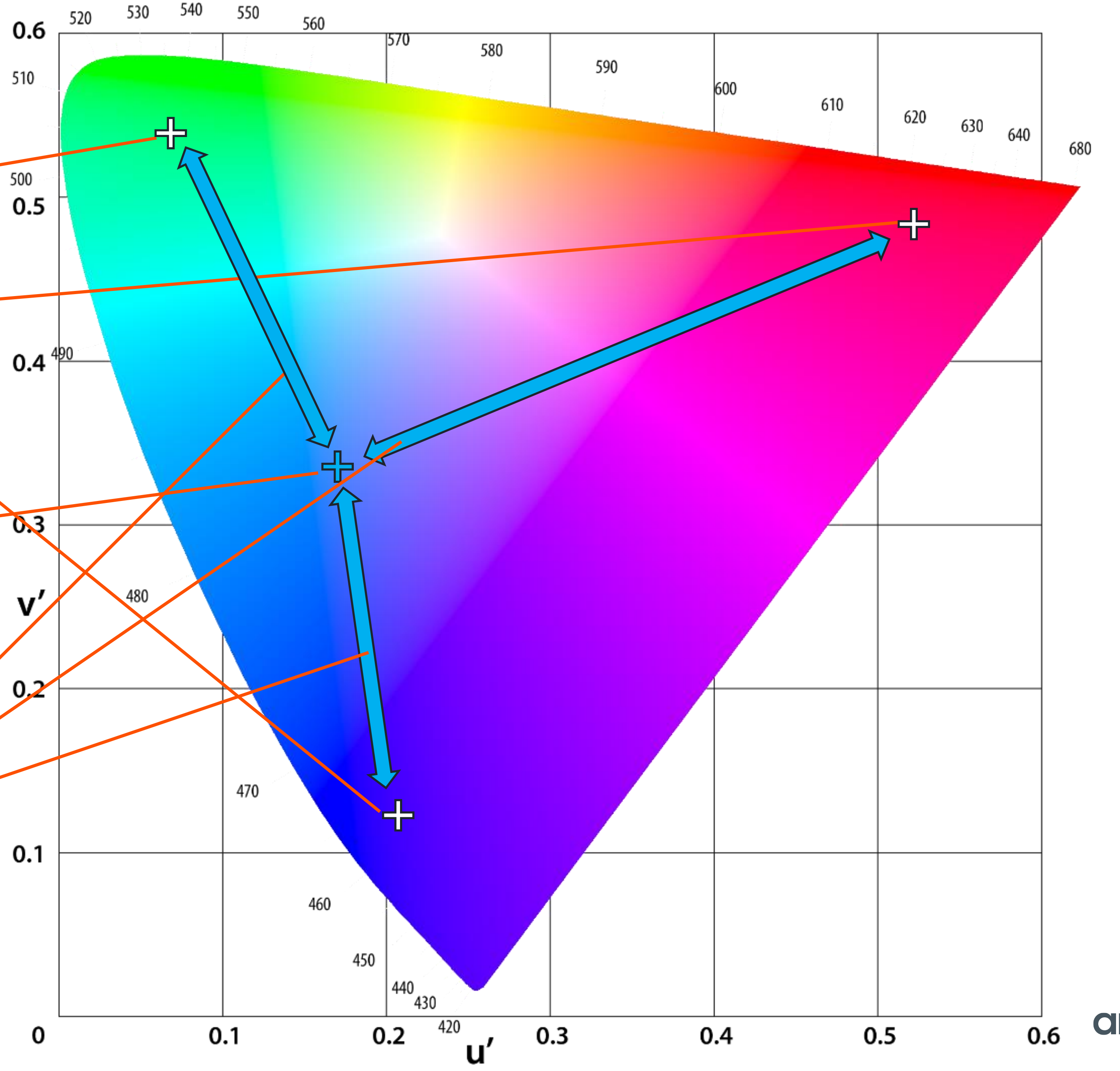
Given some physical RGB triplet, calibration finds the exact color point for the green LED

and this point for red

…and point for blue.

Assume this is the **target** color point we want show (including some I_v in Z-axis)

Then we need to compute these duty cycles to mix R/G/B to reach that target



ams OSRAM

Moving to another color space

To make computation easier

- The calibration color space is typically u', v', I_v, or C_x, C_y, I_v
- Those are good to calibrate/describe colors, but those are not well suited to compute with
- There is another color space, CIEXYZ, which has computation power.
- As Wikipedia states:
  A useful application of the CIEXYZ color space is that a mixture of two colors in some proportion lies on the straight line between those two colors.

- Therefore, we map the "calibration" color space (u', v', I_v) or (C_x, C_y, I_v)
- to the "computation" color space (X, Y, Z)
- which is often called the tristimulus vector (here named T)

$$T = \begin{bmatrix} X_T \\ Y_T \\ Z_T \end{bmatrix} = I_v \begin{bmatrix} \frac{9}{4}\frac{u'}{v'} \\ 1 \\ \frac{12 - 3u' - 20v'}{4v'} \end{bmatrix} = I_v \begin{bmatrix} \frac{C_x}{C_y} \\ 1 \\ \frac{1 - C_x - C_y}{C_y} \end{bmatrix}$$

from (u', v', I_v) to (X,Y,Z)

from (C_x, C_y, I_v) to (X,Y,Z)

Scary looking formulas, you can just call e.g. **`aomw_color_cxcyiv3_to_xyz3()`**

A2 uni col

Mixing the R, G and B

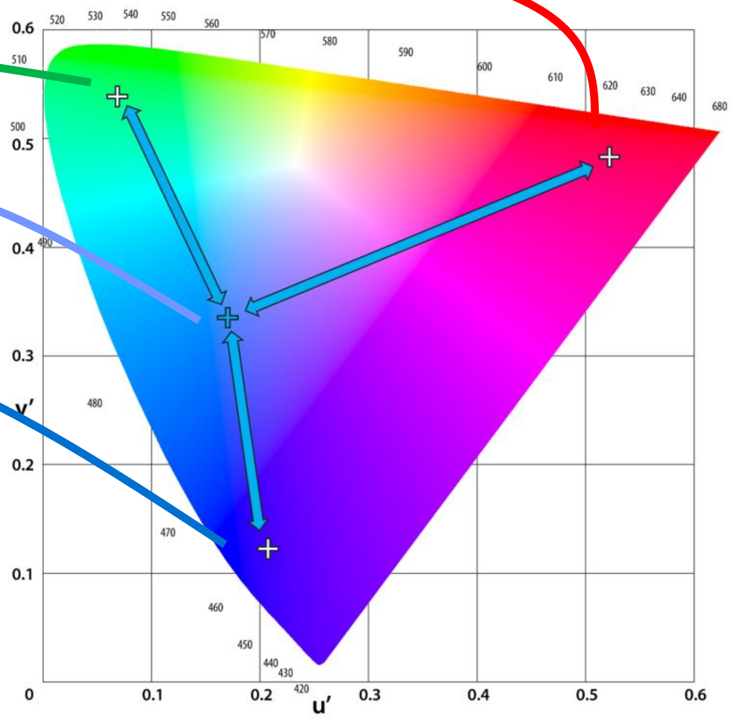
To reach a target color (X_T, Y_T, Z_T)

Tristimulus of the red LED of the triplet (from calibrating the triplet)

Green tristimulus

Blue tristimulus

Tristimulus vector of the wanted target color



$$\begin{pmatrix} X_R & X_G & X_B \\ Y_R & Y_G & Y_B \\ Z_R & Z_G & Z_B \end{pmatrix} \times \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} X_T \\ Y_T \\ Z_T \end{pmatrix}$$

Matrix multiplication computes X_T as follows

This works in the X,Y,Z color space!

$$X_T = aX_R + bX_G + cX_B$$

The problem is to find a,b,c

a is the duty cycle for red
b for green and c for blue

How to find a,b,c ?

Easy!

The problem was to find a,b,c

$$\begin{pmatrix} X_R & X_G & X_B \\ Y_R & Y_G & Y_B \\ Z_R & Z_G & Z_B \end{pmatrix} \times \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} X_T \\ Y_T \\ Z_T \end{pmatrix}$$

in matrix notation

$$A \times x = T$$

solve

$$A^{-1} \times A \times x = A^{-1} \times T$$

solution

$$x = A^{-1} \times T$$

Scary looking formula, you can just call **`aomw_color_computemix()`** and that gives you the duty cycles for the R, G, and B of the triplet (to be used in the **`setpwm`** telegram)

PWM values a,b,c (ie x) are found by inverting the tristimulus matrix of the triplet and multiplying that with the tristimulus vector of the target color

A2
uni col

Uniform color over temperature

Temperature shift

Temperature causes shift in color of the three RGB LEDs

- Each of the LEDs of an RGB is calibrated
- Calibration is at a fixed known temperature (T_{cal})

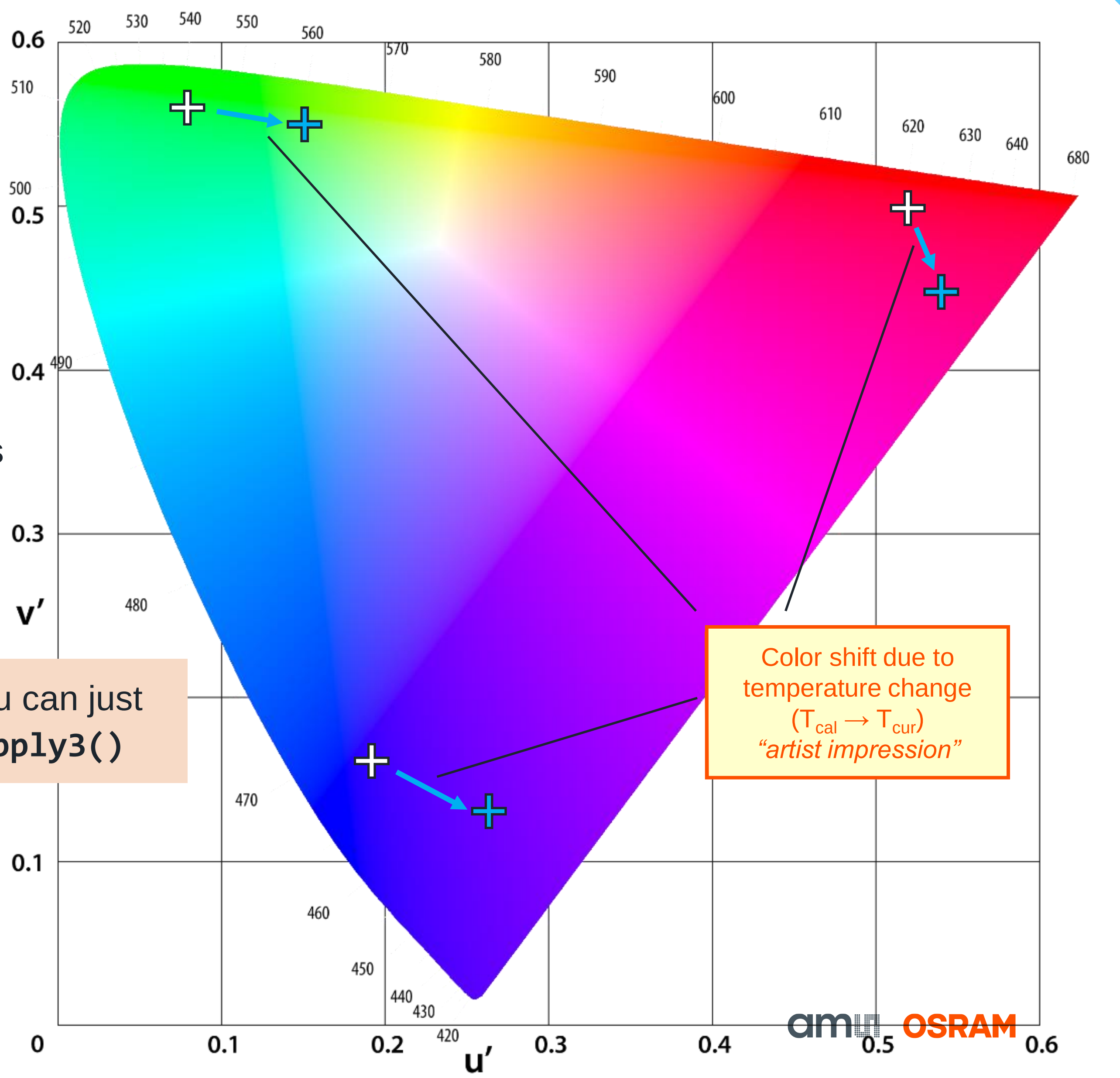
- Temperature in operation changes to T_{cur}
- Temperature change causes a shift of the colors of the 3 LEDs

- The temperature shift is $\Delta T = T_{cur} - T_{cal}$
- Then each of u', v', I_v (or each of C_x, C_y, I_v) changes as ***F***, quadratically in T:

$$\boldsymbol{F}_{\text{Tcur}} = (p\Delta T^2 + q\Delta T + 1) \times \boldsymbol{F}_{\text{Tcal}}$$

Scary looking formula, you can just call **`aomw_color_poly_apply3()`**

- The constants *p* and *q*, could be generic for a LED *type* or could be specific per instance
- but different for each of u', v' and I_v (or C_x, C_y, and I_v)



ams OSRAM

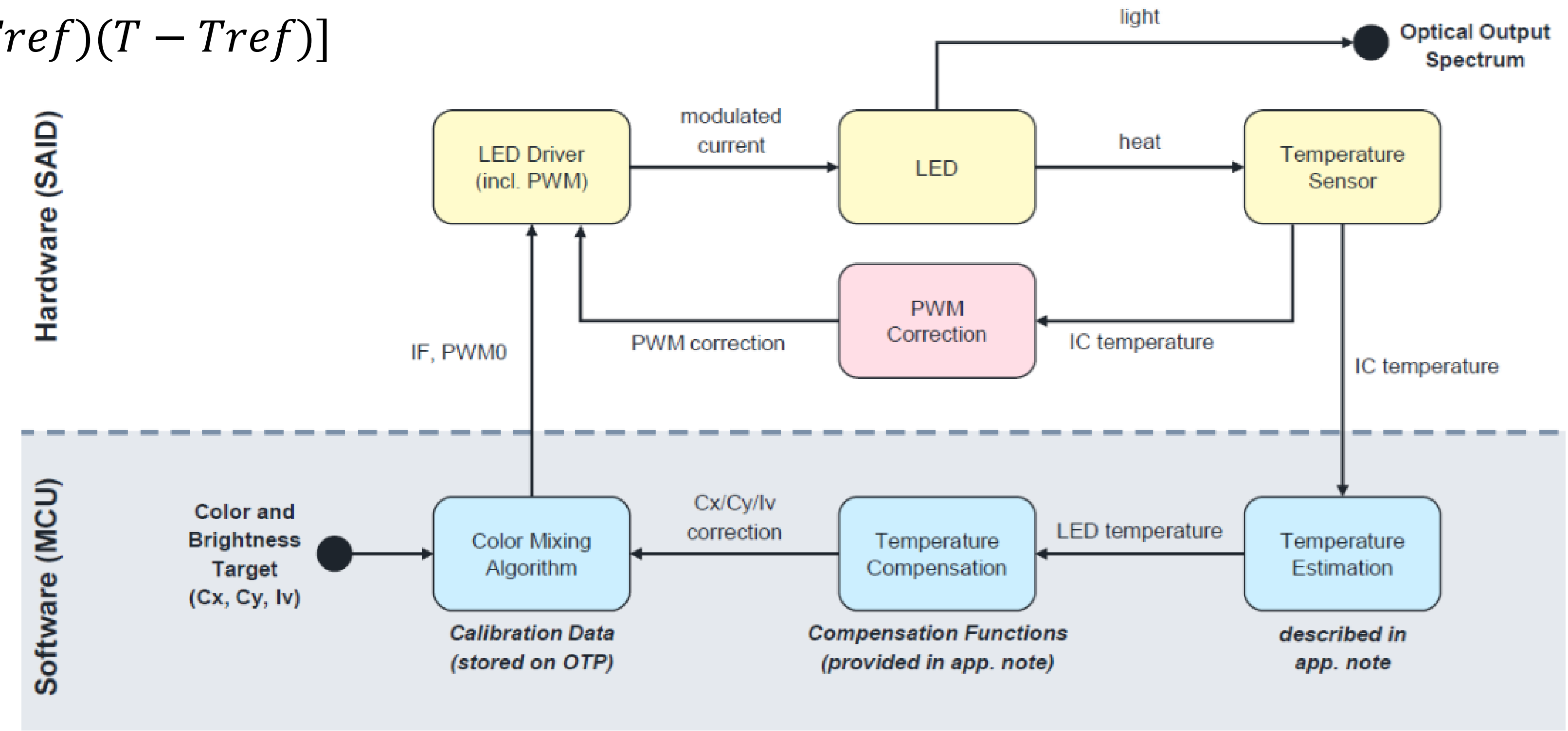
SAID's temperature compensation

A new feature to relax timing

- SAID has a feature known as temperature compensation (redish box below)
- It does a local (in-SAID, not involving RootMCU) PWM duty cycling correction for changing temperatures
- This relaxes the timing requirements imposed on the RootMCU for temperature compensation
- See section 5.6.1 LED temperature compensation in datasheet

- Formula

$$PWM(T) = PWM\,(Tref)\;[1 + \alpha\,(Tref)(T - Tref)]$$



A2 uni col

Overview

Overview

See aomw_colordemo.ino

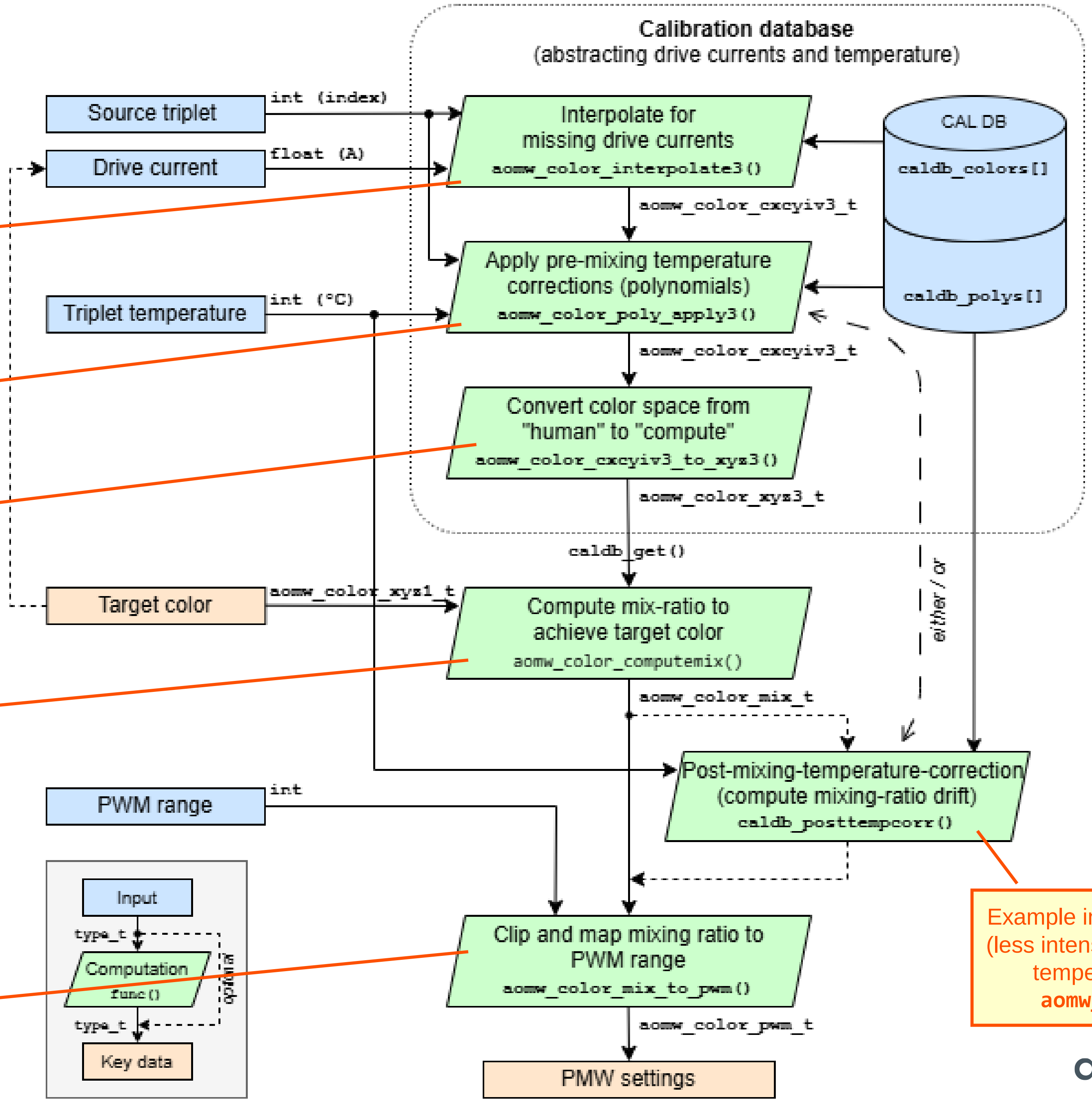
Calibration parameters are drive current specific; interpolate if the calibration data for the required drive current is not available

Temperature drift of the R, G, B LED of the triplet (see slide 10)

Convert to tristimulus domain (see slide 6)

Matrix inverse from slide 8

Convert (float) duty cycles to PWM values for SETPWMCHN



Example in **aomw_colormath.ino** (less intensive than the pre-mixing temperature correction in **aomw_colordemo.ino**)

ams OSRAM