FACULTAT
DE CIÈNCIES
I TECNOLOGIA
UVIC | UVIC·UCC

Experimental Final Degree Project

Open-source spectrophotometers: Hardware and analytical performance optimization

Josep Caballé Benavent

Degree in Industrial Electronics and Automatisms Engineering

Tutor: Mainardo Gaundenzi Asinelli

Vic, 7th of June 2021

Acknowledgements

I want to thank my thesis supervisor, Doctor Mainardo Gaudenzi Asinelli, for his guidance throughout the work, introducing me to the world of research, and the trust he has placed in me to carry out this project.

I would also like to thank my degree coordinator, Professor Juli Ordeix i Rigo, for all the help provided throughout all these years.

Finally, I would like to thank my partner for all the support she provided me through this entire journey and patience.

Abstract (Català)

Títol: Espectrofotòmetres de codi obert: Optimització analítica i de *hardware*

Autor: Josep Caballé Benavent

Supervisor: Mainardo Gaundenzi Asinelli

Data: Juny de 2021

Paraules clau: Espectrofotometria, espectròmetre, codi obert, colorimetria

En aquest treball, s'ha utilitzat un espectròmetre de codi obert C12880MA del fabricant Hamamatsu Photonics, per tal de desenvolupar un sistema capaç d'analitzar, per reflectància, dades colorimètriques de peces d'art, amb una alta precisió i repetibilitat.

Per tal d'assolir aquest objectiu, s'ha realitzat una profunda recerca en sistemes similars ja existents, i com els diferents autors d'aquests treballs han provat de millorar la qualitat de les dades obtingudes mitjançant la disminució de la *dark current* a la sortida del sistema. El treball, presenta com s'ha realitzat el desenvolupament d'un sistema que tenint en compte diversos factors: La correcta selecció d'un convertidor de senyal analògica a digital, la utilització d'un amplificador operacional, per tal de reduir el consum de corrent a la sortida del senyal analògic del sensor, i procurar disminuir l'augment de temperatura que el sensor patia, i la correcta selecció del temps d'exposició del sensor al llum reflectit de l'il·luminat sobre la superfície a analitzar, per tal d'assegurar de captar-ne tota la llum necessària pel anàlisis sense deixar-ne entrar massa, ha sigut possible millorar notablement la qualitat de les dades obtingudes.

Abstract (English)

Title: Open-Source spectrophotometers: Hardware and analytical optimitzation

Author: Josep Caballé Benavent

Supervisor: Mainardo Gaundenzi Asinelli

Date: June 2021

Key words: Spectrophotometry, spectrometer, open-source, colorimetry

In this work, an open-source spectrometer C12880MA from the manufacturer Hamamatsu Photonics, has been used. It aims to develop a robuts system capable of analyzing colorimetric data of art works with high accuracy and precission.

To achieve this goal, in-depth research has been carried out on similar existing systems and how the different authors of these works have tried to improve the quality of the data obtained by reducing the dark current at the system's output. The work presents how the development of a system has been carried out that taking into account several factors: The correct selection of an analog-to-digital signal converter, the use of an operational amplifier to reduce the consumption of current at the output of the analog signal of the sensor and try to decrease the increase in temperature that the sensor suffered, and the correct selection of the exposure time of the sensor to the reflected light of the illuminated on the surface to be analyzed, to ensure that all the light needed for the analysis is captured without letting too much in, it has been possible to improve the quality of the data obtained significantly.

Table of Contents

List of Abbreviations

ADC	Analog-to-Digital Converter
CCT	Correlated Color Temperature
CIE	Comission Internationale de l'Eclairage
CNC	Computer Numerical Control
CPU	Central Processing Unit
CRI	Color Rendering Index
CSV	Comma-separated Value
DR	Dynamic Range
I^2C	Inter-Integrated Circuit
IDE	Integrated Development Environment
LED	Light-emitting diode
MCU	Microcontroller Unit
Op-amp	Operation Amplifier
PGA	Programmable Gain Amplifier
SAR	Successive Approximation Register
SNR	Signal-to-Noise Ratio
ToF	Time-of-Flight
UAV	Unnamed Aerial Vehicle
$\Delta\Sigma$	Sigma Delta

1. Introduction

The open-source hardware approach to technology design and development is making its way among the academic community, becoming an operational model to boost knowledge sharing, cooperative approach, and innovation (Pearce, 2017a; Gaudenzi Asinelli et al., 2019; Moritz et al., 2019). During the last few years, researchers' interest on open-source hardware rapidly growth thanks to the spread of low-cost, open-source boards based on easily available and programmable microcontroller units (MCU), as well as the launch of 3D rapid prototyping and derived CNC-based machines. The availability on the market of open-source MCU-based prototyping boards such as those developed by the firms Arduino and Expressifs – to name a few, highly contributed to the design and build of analytical devices and mechatronics systems (i.e., Català et al. 2017; Pearce 2017b) that comply with scientific reliability and break the typical budget constraints that limit research advancement.

Very recently, Pearce (2020) published a review that lists 119 articles dealing with the design and development of open-source scientific hardware. Based on this list, the growing interest during the last years of the research community on open-source hardware can be summarized by the fact that most of these articles have been published between 2017 and 2020, whilst only nine between 2010 and 2016.

1.1. State of the Art

Among other instrumentation, spectrophotometric devices are attracting the interest of academic developers. Spectrophotometry is an analytical technique used to measure light in both reflection and absorption modes. Being non-destructive, it is a quite ubiquitous technique that can be used in several applications – both industrial and scientific, such as food quality assessment, environmental light monitoring, color assessment in industrial products, track changes in color appearance in cultural heritage materials such as paintings. Notwithstanding their extended use, and as the large majority of scientific devices, spectrophotometers are quite expensive instruments. Another limit is represented by analytical exclusiveness - that is, many off-the-shelf devices are usually oriented to industry market rather than research, so that adaptations and calibrations are often necessary to fulfil specific analytical requirements. Notwithstanding, hacking a proprietary device led to copyrights infringement.

The good news is that, considering that open-source hardware is becoming a worldwide emerging business model (Pearce, 2017b; Li and Seering, 2019), many top global companies that deal with software and hardware products and services (i.e., Google, Microsoft, ARM, Hamamatsu Photonics, FLIR) are already supporting the open-source vision as viable pathway for the diversification of their business. Indeed, the Japanese manufacturer Hamamatsu Photonics is one of the first international firms that few years ago started releasing blueprints of two of its most known micro-spectrophotometer sensors, namely the C12666MA and the C12880MA. The latter has rapidly become one of the most used sensors integrated in open-source spectrophotometers, to the extent that in 2016 the GetLab makers community released a breakout board that allow to use this sensor as a plug-and-play sensor to be interfaced with prototyping boards such as those developed by Arduino and Teensy (GroupGets, n.d.). Besides the breakout board, makers and engineers involved in the in the GetLab project developed a code to run the sensor by utilizing the Arduino Integrated Development Environment, thus compatible with a wide range of microcontrollers (GroupGets, 2016).

So far, the C12880MA has been integrated into several open-source spectrophotometers projects, alone or by using the GetLab breakout board. In this thesis I refer to those prototypes based on the sensor alone. Among others, Das et al. (2016) designed a handheld portable reflectance spectrophotometer to test fruit ripeness; Hoshi et al. (2018) used the sensor to measure light environment with respect to plants growth; Kim et al. (2018) developed a portable device for measuring environmental light; Sosa-Herrera et al. (2019) used the sensor in a UAV-based crop wealth assessment system; Laganovska et al. (2020) integrated the sensor into an absorbance spectrophotometer to detect chemical compounds; Sandak et al. (2020) checked the reliability of the sensor compared to more sophisticated ones to detect defects in wood by reflectance analysis.

The common conclusion of these works is that the C12880MA is a spectrophotometric sensor that allows easy integration into customizable hardware and shows performances enough good be compared to off-the-shelf devices. From the hardware side, all these prototypes share a similar architecture by connecting the sensor to an Arduino MCU or to an Arduino-based clone, being these an Arduino Uno (Kim et al., 2018; Sandak et al., 2020), an Arduino Pro Mini (Das et al., 2016), an Arduino Nano (Laganovska et al., 2020), an Arduino Leonardo Ethernet (Hoshi et al., 2018), or an Arduino clone based on the ATMega328 chip (Sosa-Herrera et al., 2019).

Besides illumination LEDs for reflectance and transmission purposes (Das et al., 2016; Laganovska et al., 2020; Sandak et al., 2020), and Bluetooth module to send data acquired wirelessly (Das et al., 2016; Yang-Soo et al., 2018; Laganovska et al., 2020), no additional electronic components were used to connect the sensor to the MCUs, with the only exception of (Hoshi et al., 2018, p.150) that used buffers between each signal line of the sensor and the MCU in order to limit potential measurement errors due to the heat generated by the C12880MA that in this specific device is utilized in continuous modality.

From the software side, all authors generally follow that released by GroupGets (2016) and furtherly modified by Versek et al. (2016), adding modifications according to the specific application field and, or to customized components such as illuminants, trigger buttons and modules for SD card. Only Kim et al. (2018) provides substantial modifications to the code by designing an equation to lower the dark current level, based on the relationship between dark current and integration time. Notwithstanding the validity of the method proposed in (Kim et al., 2018), the use of MCUs having an integrated analog to digital converter (ADC) of only 10-bits or less – as all those used in the works cited, can severely limit the acquisition of a reliable signal. Indeed, both the earlier (Das et al., 2016, pp. 5-6) and the most recent (Sandak et al., 2020, pp. 12-13) published works based on the C12880MA sensor recognized that further implementations at both hardware and software level would introduce significant improvements in the development of scientifically reliable open source based spectrophotometers.

2. Objectives

This work aims to propose hardware and software improvements to overcome all the limitations emerged from the designs and prototypes developed by other researchers and already published. Indeed, once presented to the TFG tribunal, a summary of this work will be submitted for publication in a peer reviewed scientific journal. The next chapters focus on testing the C12880MA sensor using different hardware and software configurations, with the aim to achieve a scientifically reliable device that can provide analytically accurate and precise spectrophotometric data. Additionally, this TFG aims to find an optimal device's assemblage configuration to allow non-contact and non-invasive spectrophotometric analysis in reflectance mode. These are fundamental parameters to be respected when the analyses need to be performed on especially fragile materials such as those pertaining to Cultural Heritage artefacts. Indeed, although out of the scope of this TFG, it is worth to say that the prototype developed under this work is already used within the TecnioSpring PLUS project "BioACHS – Biodegradation Assessment for Cultural Heritage surfaces and substrates", a Beta Technological Center and Mecamat research group joint project.

2.1. Research questions

The objective of this TFG is based on two main research questions:

1. To what extent is possible to optimize the work already done in open-hardware modality by other research groups (i.e., Das et al., 2016; Kim et al., 2018; Laganovska et al., 2020) to obtain more reliable data using the C12880MA spectrometer as the core sensor of the device?

Differently from previous works, the hypothesis is that by optimizing both the hardware and software the quality of the signal acquired can be improved significantly, thus increasing the reliability of the data acquired. To achieve these improvements, I had to explore different setups for hardware (i.e., MCUs, ADC, buffer amplifier), and software and firmware (i.e., PGA, data extrapolation, data interpolation).

2. To what extent it is possible to develop an open-source spectrophotometer by respecting the non-contact and non-invasive rule of reflectance spectrophotometry utilized to analyze Cultural Heritage materials?

As introduced before, when it comes to the analysis on Cultural Heritage materials' surface, non-contact and non-invasive technology is generally required. Current developed open-hardware reflectance spectrophotometers (i.e., Das et al., 2016; Sandak et al., 2020) are non-invasive, but are conceived as contact devices. To allow non-contact analyses the prototype was designed considering the geometry of the internal illuminant with respect to the sensor acquisition area and the placement of a distance sensor to avoid contacts with samples but maintaining a 2-3mm detection spot on the analyzed area.

2.2. Research development

As described in the previous sections, the aim of this work is to provide robust hardware set-up and software implementation to optimize the performances of the sensor C12880MA when used in the design and development of open-source spectrophotometric devices, with a special focus on reflectance spectrophotometry.

On the hardware side, the prototype developed is based on the ESP32 MCU which provides a faster operational speed compared to the Arduino MCUs utilized so far. Moreover, the ESP32 allows Bluetooth and or Wi-Fi communication without the necessity to add additional shields to the main hardware. This solution also simplifies prototyping processes and reduce the overall size of the final device, thus enhancing its portability. An external 16-bits ADC is used to make full use of the potential offered by the C12880MA and obtain accurate data. An additional buffer amplifier is connected to the sensor analog output before the digital conversion to help in reducing current consumption and overheating, thus obtaining a reduction of the dark noise. On the firmware side, a proper setting of the internal Programmable Gain Amplifier (PGA) of the 16-bits ADC allowed to enhance measurement accuracy and to lower dark noise. According to the illuminant choice, dark noise can be furtherly lowered by setting an appropriate exposure time.

On the software side, data accuracy was enhanced by using Python libraries that allows to define the most suitable data range according to the field of application, the extrapolation method to be applied to obtain synthetic data in case of data gaps, and the data interpolation method to be applied for unevenly spaced data (CIE167:2005; Westland et al., 2015; Wang et al., 2017).

3. Theoretical framework

The electromagnetic spectrum can be divided in wavelengths. Light is the only portion of the electromagnetic spectrum that can be detected by the sensors located in the human eyes, the photoreceptor cells (Malacara, 2011, p. 1). There is no clear border for the range of the visible light. However, in literature, it is known that it varies from 360 - 400nm to 740 - 830nm. (Malacara, 2011, p. 2; McCluney, 2014, p. 3). The Commission Internationale de l'Eclaraige (CIE), that is the international organization that, among others, define the standards for spectrophotometry and colorimetry, recommended that the range for the visible spectrum goes from 360 to 830 nm when fluorescent material is involved (CIE 15:2004) and from 380 to 780 nm when fluorescence is not involved (CIE 167:2005).

3.1. CIE 1931 color space

As the human eye possesses three types of color sensors that react to different ranges of wavelength (Color Matters, n.d.), any sensed color can be described by three variables, which are determined from the spectrum (Whetzel, 2016) using the following color matching functions

$$X = \int_{\lambda} \bar{x}(\lambda)P(\lambda)\mathrm{d}\lambda, \qquad Y = \int_{\lambda} \bar{y}(\lambda)P(\lambda)\mathrm{d}\lambda, \qquad Z = \int_{\lambda} \bar{z}(\lambda)P(\lambda)\mathrm{d}\lambda$$

Where X, Y and Z correspond to the tristimulus values, *P(λ)* to the color stimulus function of the observed light, and $\bar{x}(\lambda), \bar{y}(\lambda), \bar{x}(\lambda)$ represent the color matching functions (Schubert, 2006, p. 294) which can be observed in Figure 1.

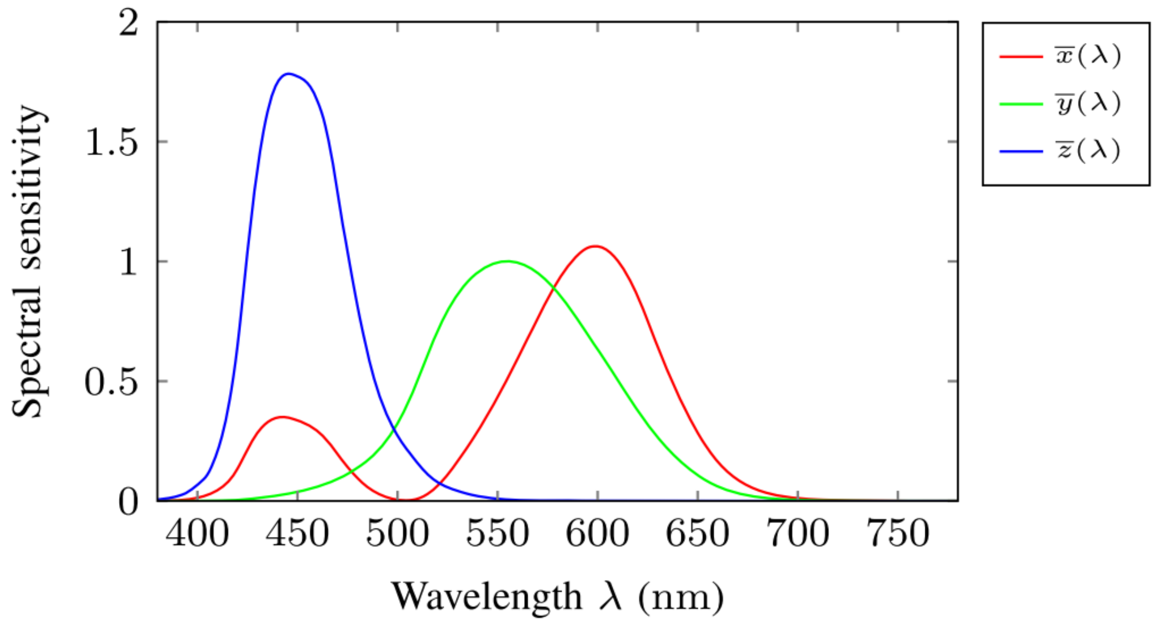

Figure 1.- CIE 1931 2º standard observer color matching functions

The color displayed by the tristimulus values can be visualized by converting the previously obtained values to chromaticity coordinates using the following equations (Poynton, 2012, p.275):

$$x = \frac{X}{X+Y+Z}, \qquad y = \frac{Y}{X+Y+Z}, \qquad z = \frac{Z}{X+Y+Z} = 1 - x - y$$

After being converted to chromaticity coordinates, the color can be seen on the CIE 1931 color space chromaticity diagram (Figure 2). The horseshoe-shaped chromaticity diagram shows all the colors that can be perceived by the human eye according to a specific viewing angle of 2 degree.

On the boundary of the diagram, monochromatic colors can be found, each showing a dominant wavelength. Colors within the diagram body (the gamut), can be obtained by mixing two other colors, either monochromatic or not (Schubert, 2006, p.295)

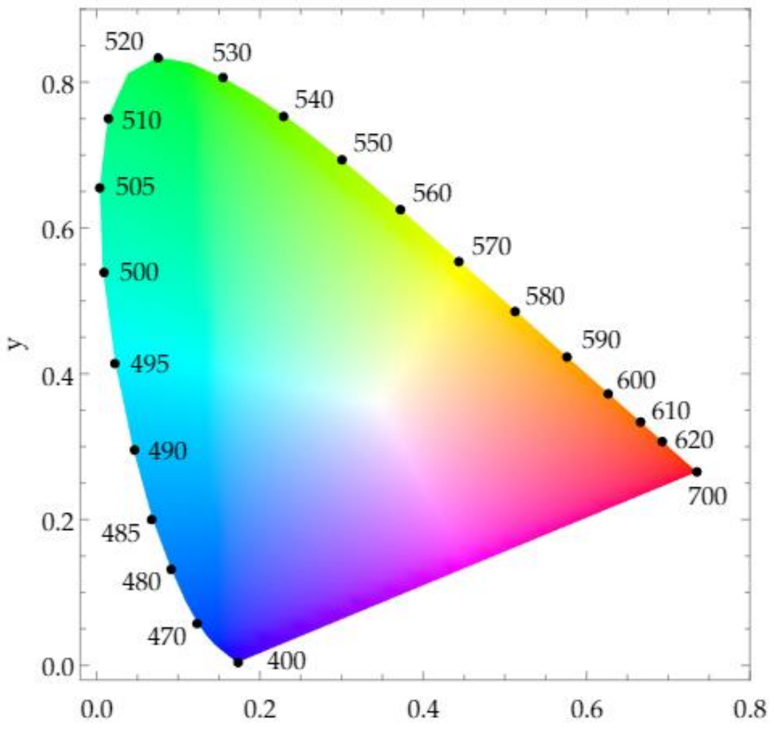

Figure 2.- CIE 1931 Chromaticity diagram.

3.2. Color Temperature

The temperature resultant from the light of a color equivalent to the source light that radiates from an ideal Planckian black body radiator is known as the color temperature. At temperatures lower than 5000K the light is called “warm” (yellowish), and at temperatures above this value “cool” (bluish) as shown in Figure 3:

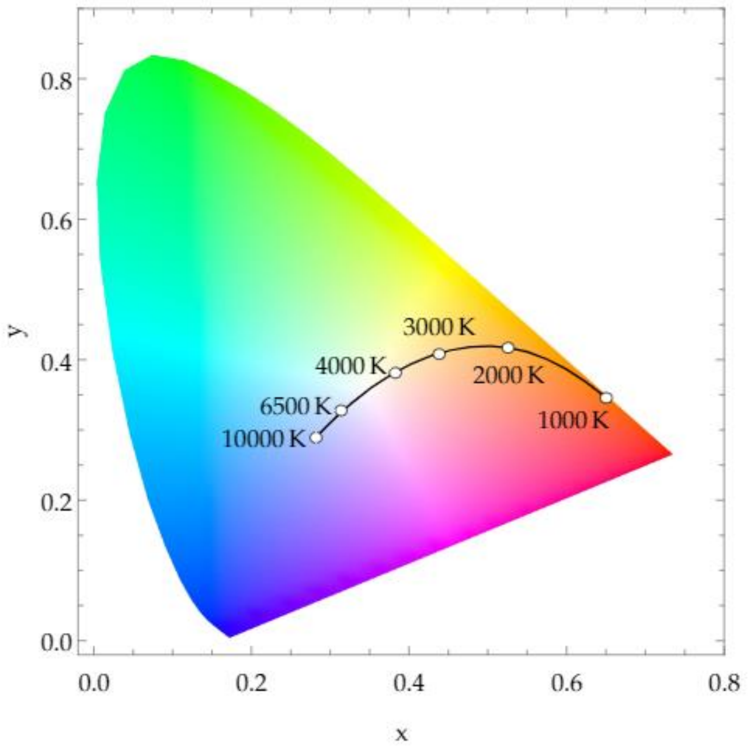

Figure 3.- CIE 1931 chromaticity diagram with a Planckian locus representing different temperature values.

To determine the color temperature of a white illuminant, the chromaticity coordinates must be calculated. Then, it must be determined to which point of the Planckian locus do the coordinates correspond. If the obtained chromaticity coordinate does not match with a valid coordinate of the Planckian locus, the temperature of the Planckian black body radiator which is closest to the color of the illuminant is used instead (Schubert, 2006, pp. 306-311).

3.3. Color Rendering Index

The spectrum of light influences the capacity to show the true colors of bodies. Two lights that might seem the same can have different spectrums. It is considered that they vary in their color rendering, and these differences are measured in terms of the color rendering index (CRI).

As a Planckian black body has a rendering index of 100 out of 100, the light reflected form this can be used as a reference light to assess the color rendering ability of an illuminant. Both lights, the reference light and the one that is meant to be tested, are used to illuminate any of the eight CIE standardized color samples proposed by Nickerson (1960) in Table 1, as specified by CIE (2004). Then, the CRI is calculated by determining how much the perceived color sample changes when it is illuminated by both sources. If no changes can be observed, it is stated that the illuminant has a CRI of 100. Otherwise, if it changes, the CRI is lowered, receiving lower CRI values for more significant changes in the perception of the color samples. (Schubert, 2006, pp. 315-324).

Name	Appr. Munsell	Appearance under daylight	Swatch
TCS01	7,5 R 6/4	Light greyish red	🟥
TCS02	5 Y 6/4	Dark greyish yellow	🟫
TCS03	5 GY 6/8	Strong yellow green	🟩
TCS04	2,5 G 6/6	Moderate yellowish green	🟩
TCS05	10 BG 6/4	Light bluish green	🟦
TCS06	5 PB 6/8	Light blue	🟦
TCS07	2,5 P 6/8	Light violet	🟪
TCS08	10 P 6/8	Light reddish purple	🟪

Table 1.- Test color samples, including their approximate Munsell notations (CIE 2004)

If no reference light is available to be used, the reflectance spectrums of the samples can be found in CIE (2004) with their approximate Munsell notations listed aside.

4. Materials and methodology

As introduced in the first chapter, this thesis has been developed under the umbrella of the open-source paradigm, and as a litmus test it uses the set-up and results of authors that developed open-source spectrophotometers based on the C12880MA sensor (i.e., Laganovska et al., 2020; Das et al., 2016; Sandak et al., 2020; Kim et al., 2018). As for that, only open-source resources were selected. Indeed, open-source software development environments such as the Jupyter Notebook and the Arduino IDE allow to use a noticeable number of open-source programming libraries that can simplify the development of projects, and promote new research designs (Pérez and Granger, 2012).

4.1. Materials

The sensor used as the central core of the device is the C12880MA developed by the Hamamatsu, a mini spectrophotometer having a size of 20.1 x 12.5 x 10.1 mm, a CCD line sensor made of 288 pixels, and a spectral response range between 240 and 850 nm with a typical resolution of 15 nm. The wavelength information is obtained by applying a 5*th* order polynomial equation (unique for each C12880MA) to the light signal detected by each of the 288 pixels.

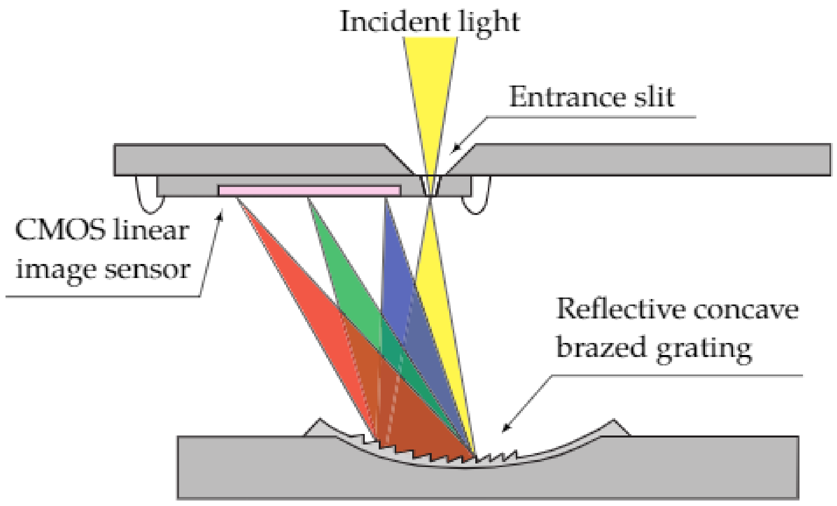

Figure 4.- C12880MA Optical component layout (Hamamatsu, 2021)

The reasons why this sensor was selected are various: Hamamatsu provides all the necessary information for the sensor to be used as an open-source component; as it does not require a constant clock pulse, the range of the MCUs that may be used in the device is not limited; a basic code example is provided open-source for using the sensor with the Arduino IDE, and another script to transform the data obtained with Python (Groupgets, 2016).

On the first step, three different open-hardware MCUs were chosen and tested: Arduino Nano, which is based on the ATmega328 and offers an 8-bits ADC; Arduino Mega 2560, which is based on the ATmega2560 and offers a 10-bits ADC; ESP32-DevKitC, which is equipped with an ESP32-WROOM-32 and offers a 12-bits internal ADC and an integrated Bluetooth and Wi-Fi module for wireless connection.

Figure 5.- From left to right: Arduino Nano ATmega328, Arduino Mega 2560 ATmega2560, and ESP32-DevKitC. (Arduino, n.d.; Espressif, n.d.)

All three of them have an operating voltage of 5V and offer a 3.3V output, which is required to run the C12880MA sensor, and offer an I2C communication interface, which simplifies the code required to run the sensors required for the prototype. Even more, these MCUs are all coded via the Arduino IDE, an open-source integration development environment which allows easy implementation of code. As can be seen in the datasheets of each MCU, the main difference between the three MCUs shown in Figure 5, besides their different internal ADC, is that only the ESP32 family has an integrated Bluetooth and Wi-Fi module. This simplifies the design of a prototype that requires wireless communication. Moreover, the ESP32 family offers an ESP32-Cam board that integrates a micro camera, expected to be used in future prototypes or devices.

The used illuminant on the prototype is a Lumileds Luxeon 3535L HE Plus (Round LES) with a nominal CCT of 6500K and a 70 CRI, which spectrum corresponds approximately to the spectrum of a LED-B5 illuminant, as defined by the CIE (CIE15:2018). The white point of both illuminants is close one to each other, being 0.3123x 0.3282y for the used illuminant (Lumileds, 2021, p.21), and 0.3118x 0.3236y for the reference illuminant LED-B5 (Wikiwand, n.d.). Additional specification of the used illuminant is that the LED's typical forward voltage is 2.9V, and typical forward current 200mA, being compatible with the selected MCUs (Lumileds, 2021, p. 5).

Moreover, to ensure a correct measurement distance between the surface to analyze and the device, a VL53L1x Time-of-Flight distance sensor is used. The VL53L1x offers a measurement range of up to 400cm, and has an operating voltage of 3.3V, making it suitable for all three MCUs.

4.2. Methodology

Many tests on several MCUs and ADCs, as well on different exposure times - that is the amount of time the detector is reached by the reflected light, need to be performed to assess the reliability of the spectrophotometric sensor, its accuracy, and its precision, and thus the best hardware configuration.

As for reflectance spectrophotometry, the geometry of the incident light – that is the light that illuminate the scene, is an important parameter to consider. Thus, the distance of the illuminant from the scene and the angle of illumination, as well as the distance of the sensor from the analyzed area, need to be carefully controlled. This is even more true when the analyst needs to perform comparative tests. To perform geometrically homogeneous tests for each hardware and software configuration a small prototype case was designed by using PTC Creo Parametric 7.0.1.0 and printed with the BCN3DSigma 3D printer using black ABS to avoid environmental light interferences. The prototype case integrates the Lumileds Luxeon 3535L HE Plus LED and the VL53L1x distance sensor.

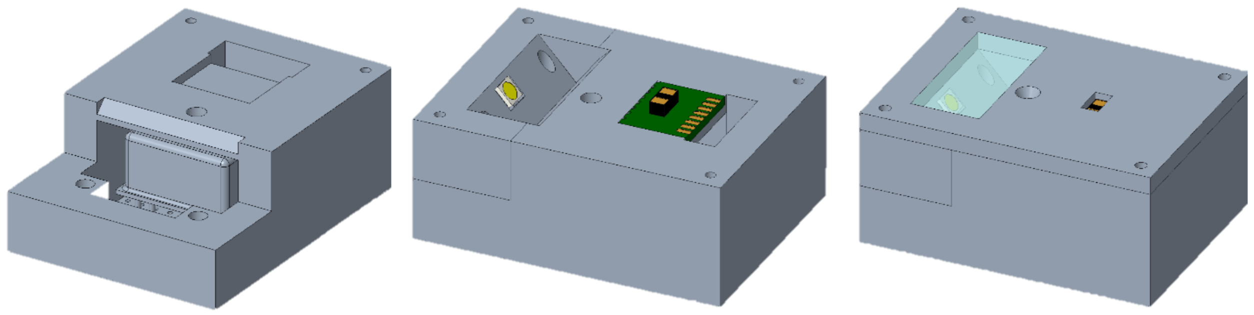

Figure 6.- Three-parts prototype case (44.1 x 38.4 x 23 mm) integrating the C1288MA sensor, a Lumileds Luxeon 3535L HE Plus, and a VL53L1X Time-of-Flight distance sensor. Additionally, a filter is used to homogenize the light from the illuminant.

Once the prototype case was designed, printed, and assembled, all three MCUs described in the previous section were tested to check the reliability of the signal acquisition. The tests were performed by using the MCUs with their internal ADC and by using an external ADC, the ADS1115. This is a 16-bits ADC, 15 of them for measuring and the last one for the sign; it uses I2C as the standard connection, and has an internal PGA, from x2/3 up to x16 to either help boost up smaller signals or shrink bigger ones.

From the firmware side, the adjustment of parameters that are required to run the sensor and acquire the electromagnetic signal, such as the exposure time according to the incident light typology, intensity, and angle, were performed to reduce the impact of the dark noise at the output. The higher the exposure time, the higher the impact the dark noise will have on the output of the system (Joula, 2017). Thus, the exposure time was modified to determine which is the minimum possible exposure time to efficiently detect the reflected light and avoid dark noise as much as possible. Additionally, a software filter was also applied to smooth the spectrum obtained without losing any relevant data.

Tests were performed using the prototype case in Figure 6 with all possible configurations. In the first test, two patches from the ColorChecker® classic 2014 by X·Rite were used: the white patch, to test if the prototype was working properly, as it should reflect the known spectrum of the illuminant used; and the black patch, to check the impact of each configuration on the dark noise at the output. Once the best configuration was selected after eliminating the dark noise, three more patches were used: red, green, and blue. At this stage of the tests, different band-with filters were used to check if the precision of the analysis could be improved by eliminating leakage of other lights different from the one expecting to be obtained, for example, the leakage of green while observing blue light.

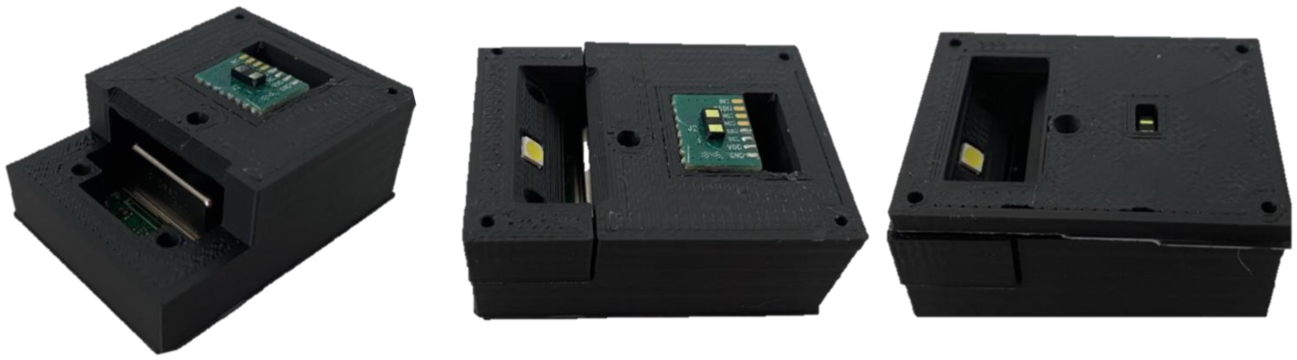

Figure 7.- Printed prototype case for tests

5. Hardware and Software Development and Tests

This section presents the experimental process followed to select the hardware for the final prototype. All the tests done to support the decisions taken are shown and explained in detail for the reader's understanding and future replication.

5.1. Systems' requirements

To lower as much as possible the dark noise at the output of the system to obtain robust data, it is important to consider that when there is a high current consumption at the video circuit of the sensor, its temperature increases, thus causing an increment of the dark noise (Hamamatsu, 2021, p.3). The analog output must be connected to a buffer amplifier before being converted to a digital signal to reduce the current flow at the video output channel of the sensor.

By observing the timing chart for the C12880MA (Figure 8) and the Arduino script (Groupgets, 2016), it could be determined that one measurement cycle of the sensor consists of two periods: the integration time and the output.

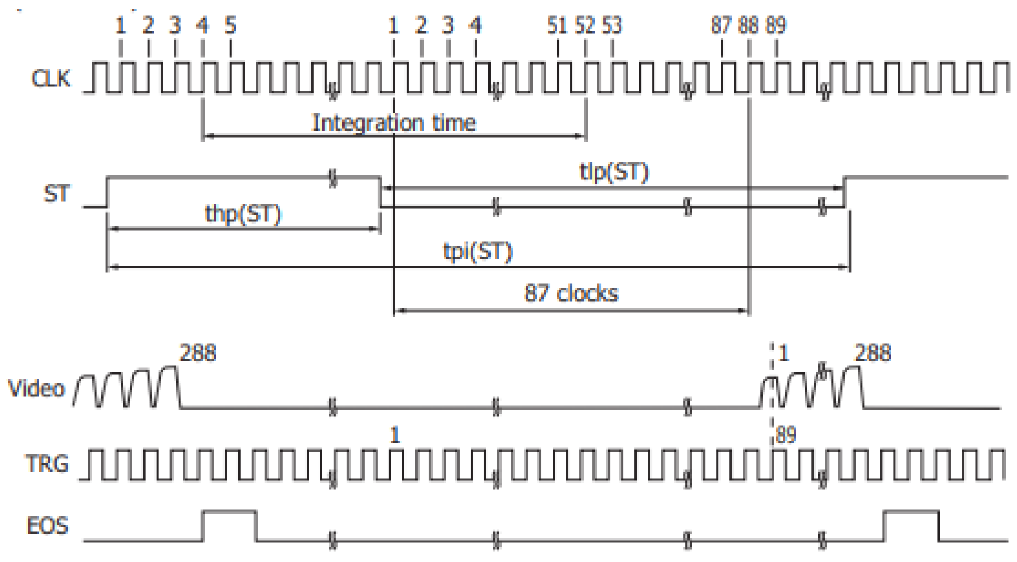

Figure 8.- Timing chart of Hamamatsu C12880MA. (Hamamatsu, 2021, p.7)

Throughout the integration time period, the incident light from the LED reflected by the analyzed surface acquired. That period is equal to the high period of start (ST) pulse plus 48 cycles of clock pulses (Hamamatsu, 2021, p. 8). These 54 cycles (48 cycles plus 6 cycles corresponding to the minim high period of ST) lead to a minimum integration time of 10.8 while the sensor is clocked at 5 MHz. Therefore, the minimum possible exposure time of the sensor to the reflected light is equal to the minimum integration time. And the lower the exposure time, the lower the dark noise at the output of the system (Kim et al., 2018).

Additionally, to determine which would be the minimum effective resolution of the ADC to convert the pixel data accurately, the values of the saturation output voltage and the readout noise (*Hamamatsu, 2021,* page 2) are used to calculate the dynamic range (DR) of the sensor. The DR is defined as the highest achievable signal level divided by the noise level when no light is detected by the spectrometer (Ibsen Photonics, n.d., page 1), and is calculated as follows:

$$DR = 20 * log_{10}\left(\frac{V_{sat} + V_0}{Nr}\right) = 20 * log_{10}\left(\frac{4.8V}{1.8mV_{RMS}}\right) = 68.51\ dB$$

where DR stands for dynamic range, V_{sat} for the saturation voltage, V_0 for the offset voltage, set to 0.5V, and Nr for the readout noise. The obtained dynamic range in dB (base-10), can also be read as a 2667:1 (ratio), or a 11.4-bit resolution (base-2) (Technote, 2011).

Assuming this value, means that an ADC with a resolution of at least 12 bits to obtain accurate data from the sensor's video output. However, for marketing reasons, as there is a limited range of commercial 12-bit ADCs, a 16-bit ADC was selected. This parameter, as well as the fact that the sensor had to be clocked to 5 MHz to work at its minimum possible integration time, thus reducing the dark noise of the system, and that a MCU with a data rate transmission of at least 80 Mbps is required, reduces the list of available open-source MCUs to just a few.

Therefore, by summing up all the above-mentioned information, it could be determined that the system would require specific components: 1) an ADC, either internal or external, with a resolution of 16 bits to obtain accurate values for the pixel data, 2) a buffer amplifier, to reduce the current consumption at the video circuit of the sensor, thus reducing the dark noise, 3) an exposure time of the sensor with respect to the reflected light from the analyzed spot low as much as possible, 4) and a MCU able to provide a high clock pulse frequency in the above-mentioned conditions.

5.2. Software for tests

Two different scripts were used to perform all the analysis and tests required for the design of the device: one that runs on the MCU via the Arduino IDE to obtain the raw data from the sensor, and another that runs on the CPU with Jupyter Lab, using Python, to interpret the data, plot it, and analyze it. Both scripts are described in detail in this subsection, and future modifications of the software are referenced here.

5.2.1. Firmware running on the Microcontroller Unit (MCU).

For further information and steps on how to perform the actions mentioned in this subsection, and for full software scripts and download links, check GroupGets (2016).

To be able to use the external ADS 1115 and the Time-of-Flight sensor VL53L1X, it is necessary to upload the libraries pertaining to both components from the Arduino IDE library manager. Once uploaded, the libraries are included in the script as follows:

```
#include <Wire.h>                                                             1
#include <Adafruit_ADS1015.h>    //Required for the external ADC ADS1115
#include <VL53L1X.h>             //Required for the ToF Sensor VL53L1X
```

The second step consists in defining which pins from the sensor are used, and to which pins of the MCU these are connected. The same definitions apply to other components used, such as the illuminant and the status LEDs. For other components using I^2C communication this step is not required:

```
// Hamamatsu C12880MA Macro Definitions                                   2
#define SPEC_TRG         A1      //Output Signal
#define SPEC_ST          A2      //Input Signal
#define SPEC_CLK         A3      //Output Signal
#define SPEC_VIDEO       A4      //Comment if using an external ADC ADS1115

#define SPEC_CHANNELS    288     //C12880MA's number of pixels

// Reading LED Definition
#define LED_WHITE        12

// Status LEDs Definitions
#define LED_GREEN        11
#define LED_RED          10
```

Required global constants and variables are then defined as follows:

```
Adafruit_ADS1115 ads;            //For simplification                     3
VL53L1X tof;                     //For simplification

uint16_t data[SPEC_CHANNELS];    //Pixel data is stored in this variable
float Voltage = 0.0;             //Used to determine voltage for each pixel data
float vol[SPEC_CHANNELS];        //For error detections using above variable
```

Once the pins are defined, these are automatically set as inputs. In the void setup of the script it is important to identify the pins that are outputs. Also, the serial port is defined alongside other elements required for the setup of the system, such as the gain of the PGA, the ADC, and the TOF sensor. To set up correctly all the elements of the C12880MA sensor, it is important to use the information from the timing chart in Figure 8. That is why it is also required to set the clock as high and start as low.

```
void setup(){                                                                          4

  //Set desired pins to OUTPUT
  pinMode(SPEC_CLK, OUTPUT);
  pinMode(SPEC_ST, OUTPUT);
  pinMode(LED_WHITE, OUTPUT);
  pinMode(LED_GREEN, OUTPUT);
  pinMode(LED_RED, OUTPUT);

  digitalWrite(SPEC_CLK, HIGH);     //Set SPEC_CLK High
  digitalWrite(SPEC_ST, LOW);       //Set SPEC_ST Low

  Serial.begin(115200);             //Baud Rate set to 115200

  // Uncoment one of the following lines
  ads.setGain(GAIN_TWOTHIRDS);     //+/- 6.144V  1 bit = 0.1875mV (default)
  // ads.setGain(GAIN_ONE);        //+/- 4.096V  1 bit = 0.125mV
  // ads.setGain(GAIN_TWO);        //+/- 2.048V  1 bit = 0.0625mV
  //ads.setGain(GAIN_FOUR);        //+/- 1.024V  1 bit = 0.03125mV
  // ads.setGain(GAIN_EIGHT);      //+/- 0.512V  1 bit = 0.015625mV
  // ads.setGain(GAIN_SIXTEEN);    //+/- 0.256V  1 bit = 0.0078125mV

  Wire.begin();
  Wire.setClock(400000);           //Use 400 kHz I2C
  ads.begin();

}
```

From this point on, the measurement cycle of the sensor, which consists of an integration and an output part, is finally coded (see subsection 5.1. System's requirements). This part of code related to the integration time was developed by GroupGets (2016), and it has been proved that works

```
void readSpectrometer(){                                                        5

  int delayTime = 1;                    // delay time
  int16_t SPEC_VIDEO;


  // Start clock cycle and set start pulse to signal start
  digitalWrite(SPEC_CLK, LOW);
  delayMicroseconds(delayTime);
  digitalWrite(SPEC_CLK, HIGH);
  delayMicroseconds(delayTime);
  digitalWrite(SPEC_CLK, LOW);
  digitalWrite(SPEC_ST, HIGH);
  delayMicroseconds(delayTime);

  //Sample for a period of time

  for(int i = 0; i < 15; i++){
      digitalWrite(SPEC_CLK, HIGH);
      delayMicroseconds(delayTime);
      digitalWrite(SPEC_CLK, LOW);
      delayMicroseconds(delayTime);

  }

  //Set SPEC_ST to low
  digitalWrite(SPEC_ST, LOW);

  //Sample for a period of time
  for(int i = 0; i < 85; i++){

      digitalWrite(SPEC_CLK, HIGH);
      delayMicroseconds(delayTime);
      digitalWrite(SPEC_CLK, LOW);
      delayMicroseconds(delayTime);

  }

  //One more clock pulse before the actual read
  digitalWrite(SPEC_CLK, HIGH);
  delayMicroseconds(delayTime);
  digitalWrite(SPEC_CLK, LOW);
  delayMicroseconds(delayTime);
```

Up to this point, the code for the integration time part starts, but still do not read any value from the sensor. On the following script 6, the values from each pixel are read alongside with the correspondent voltage values for each one of them. The values are then stored in their respective variables for accessing them later.

6

```
//Read from SPEC_VIDEO
  digitalWrite(LED_WHITE, HIGH);       // Turn on the white LED
  delayMicroseconds(500);

  for(int i = 0; i < SPEC_CHANNELS; i++){

      //SPEC_VIDEO = ads.readADC_SingleEnded(1);  //Uncoment when using ADS1115
      data[i] = SPEC_VIDEO;
      Voltage = (SPEC_VIDEO * 0.03125)/1000;
      vol[i] = Voltage;

      digitalWrite(SPEC_CLK, HIGH);
      delayMicroseconds(delayTime);
      digitalWrite(SPEC_CLK, LOW);
      delayMicroseconds(delayTime);

  }

  digitalWrite(LED_WHITE, LOW);         // Turn off the LED after reading

  //Set SPEC_ST to high
  digitalWrite(SPEC_ST, HIGH);

  //Sample for a small amount of time
  for(int i = 0; i < 7; i++){

      digitalWrite(SPEC_CLK, HIGH);
      delayMicroseconds(delayTime);
      digitalWrite(SPEC_CLK, LOW);
      delayMicroseconds(delayTime);

  }

  digitalWrite(SPEC_CLK, HIGH);
  delayMicroseconds(delayTime);

}
```

The second part of the measurement cycle, which consists of printing the values stored on the data array, is a simple *for loop* that prints each value of the array one by one, as follows:

7

```
void printData(){

  for (int i = 0; i < SPEC_CHANNELS; i++){

    Serial.print(data[i]);
    Serial.print(',');

  }

  Serial.print("\n");

  for (int i = 0; i < SPEC_CHANNELS; i++){

    Serial.print(vol[i]);
    Serial.print(',');

  }

  Serial.print("\n");
}
```

Finally, the program is run.

8

```
void loop(){

  readSpectrometer();
  printData();
  delay(10);

}
```

5.2.2. Software running on the Central Processing Unit (CPU).

The software required for the data post-processing, plotting, and storage is run in Python through the Jupyter Lab. This script also allows modifying the integration time of the sensor. It is run on the CPU of the system without requiring any drivers to be installed. As Jupyter Lab is an open-source IDE that most people use to develop projects that follow the open-source paradigm, it is easy to find libraries developed by other users all around the web that would simplify the code a lot. In order to use those libraries, it was required to import them as follows:

```
import matplotlib.pyplot as plt                                          9
import numpy as np
import pandas as pd

import os, serial, time

from pprint import pprint

from sklearn import preprocessing
from sklearn.preprocessing import MinMaxScaler
from scipy.signal import lfilter

import colour
```

The first three libraries in script 9 are the ones used for the most general purposes: *numpy*, aka numerical Python, is the most known Python library that allows work with arrays and was developed by Travis Oliphant in 2005; *matplotlib*, is a plotting library which provides an object-oriented API for embedding plots into applications and was developed by John D. Hunter in 2003; *pandas*, which is a software library for data manipulation and analysis developed by Wes McKinney in 2008. The libraries listed in the fourth import line -the *os*, *serial*, and *time* libraries, are used to enable serial communication between the CPU and the MCU. The *pprint* library is a helpful module that allows the user to "pretty-print" arbitrary Python data structures in a form that can be used as input to the interpreter. This is one of the most important libraries when trying to understand the capabilities of a new library and was developed by Fred L. Drake alongside a small group of contributors through Github. The *sklearn* and *scipy* libraries are used for data manipulation and visualization. These are used in order to obtain normalized data and apply a filter to smooth the obtained signals.

Finally, the *colour* library is an incredibly powerful library for spectrophotometry. It allows several kinds of interpolation in order to convert the obtained data in a internationally standardized way, as defined by the Commission International de l'Eclaraige (CIE), and it allows to visualize the data acquired in the desired color space (i.e., CIE XYY, CIE L*a*b), besides many other actions that were not used during the development of the project. *Colour* is affiliated to NumFocus, a 501(c)(3) nonprofit in the United States.

Once the libraries are imported, the serial communication between the CPU and the MCU can be enabled. The serial communication is done by object-oriented programming as shown in script 10:

10

```
class MicroSpec(object):
    def __init__(self,port):
        self._ser = serial.Serial(port, baudrate = 115200)

    def set_integration_time(self, seconds):
        cmd = "SPEC.INTEG %0.6f\n" % seconds
        self._ser.write(cmd.encode('utf8'))

    def read(self):

        self._ser.write(b"SPEC.READ?\n")
        sdata = self._ser.readline()  #Read the serial port as it is - string
        sdata = sdata.split(b",")
        sdata.pop()                   #Erase the last byte (b"\n")
        sdata = np.array([int(p) for p in sdata])

        self._ser.write(b"SPEC.TIMING?\n")
        tdata = self._ser.readline()   #Read the serial port as it is - string
        tdata = tdata.split(b",")
        tdata.pop()                    #Erase the last byte (b"\n")
        tdata = np.array([int(p) for p in tdata])

       #Save the value to access it when calling function spec.read()
        return(sdata, tdata)

port = ("/dev/tty.SLAB_USBtoUART")     #Determine the port used to access the MCU

spec = MicroSpec(port)
icc = spec.set_integration_time(100e-3)#Define the Integration Time
```

Once the serial communication port and baud rate are set, the integration time can be defined by simply changing the time value from the last line of the code of script 10. Lastly, the class read the values obtained values via the software run on the MCU as a comma-separated value string, which is then written in an array to eliminate the last character, corresponding to “\n”, leaving the resultant 288 strings obtained from each pixel, which are consequently converted to integers.

By returning those two arrays, of 288 integer values, when calling the class, it is easier to access those values at any moment of the code.

Hamamatsu requires each user to calibrate its own C12880MA spectrometer, depending on its serial number, by providing a five-degree polynomial that need to be computed to obtain the exact wavelength values, in nm, according to the pixel response. For the one used in this project, the given values were the ones shown in the following script 11, which was basically used to compute the polynomial and obtain a 288-integer values array.

11

```
pix=arange(1,289);

A_0=3.059724361e+2; B_1=2.716608005; B_2=-1.381038385e-3;
B_3=-5.227858411e-6; B_4=-1.117311012e-9; B_5=2.023350975e-11;

nm=A_0+B_1*pix+B_2*pix**2+B_3*pix**3+B_4*pix**4+B_5*pix**5;
x-axis = nm;

sdata, tdata = spec.read()
y-axis = sdata;

spectrum = (x-axis,y-axis)
```

With the two arrays stored and merged into a 2-D array, each dimension representing one of the axes of the plots, the data can be plotted. The plot can be used to quickly visualize the data and determine how different configurations are affecting the quality of the output before saving the data in CSV files for future post-processing.

12

```
plt.figure(1)                                               #First plot

plt.plot(x-axis,y-axis)

plt.title('Spectral data')                                  #Plot's title
plt.legend(('sample 1',), frameon=False)                    #Plot's legend
xlabel('Wavelength(nm)')              #Label x-axis as Wavelengngth(nm)
ylabel('Spectral intensity (ADU)')    #Label y-axis as ADU = Analog/Digital Unit
```

However, with the plot obtained in script 12, it is not possible to perform reliable comparisons when testing different hardware configurations, as the data are not normalized. So, the next step is to normalize the data and plot it again before saving these as a CSV file for comparisons.

```
xlist = x-axis.tolist()                                                    13
y = y-axis.tolist()

ylist = np.reshape(y, (-1, 1))

scaler = MinMaxScaler()
scaler.fit(ylist)
normalized = scaler.transform(ylist)
normalized1d = normalized.flatten()

plt.figure(2)
plt.plot(xlist, normalized1d)

CSVfile = np.stack([newxlist,normalized1d])
np.savetxt(file.csv', CSVfile, delimiter = ',', header = "Wavelenght(nm),Spectral
intensity (ADU)")
```

By using the *sklean* library's preprocessing tool *MinMaxScaler*, it is possible to normalize all data between 0 and 1, then plot and save these in a CSV file for later comparisons with other data in order to determine differences or similarities.

The last part of the code is focused on obtaining the tristimulus values for each spectrum obtained. To do this, the *colour* library was used:

```
zipbObj = zip(newxlist, ylist)                                             14
rdata = dict(zipbObj)           # Create a dictionary from zip object

sd = colour.SpectralDistribution(rdata, name='Sample')
print(repr(sd))
cmfs = colour.CMFS['CIE 1964 10 Degree Standard Observer']
illuminant = colour.ILLUMINANTS_SDS['D65']
```

Script 14 allows transforming the data obtained from the C12880MA to a spectral distribution. It is mandatory to define which CIE-defined standard observer is used, either the CIE 1931 2 Degree or the CIE 1964 10 Degree standard observers, and which illuminant is used.

Once the data were transformed to a spectral distribution format valid for processing it with the *colour* library to obtain the tristimulus values, the data are interpolated in order to obtain constantly separated wavelength values t an interval of 10nm, as required by the CIE (Wang et al., 2017). Then, the tristimulus values can be obtained, as shown in script 15:

```
sd_copy.interpolate(colour.SpectralShape(400, 800, 10),                          15
interpolator=colour.CubicSplineInterpolator)

# Calculating the sample spectral distribution *CIE XYZ* tristimulus values.
XYZ = colour.sd_to_XYZ(sd_copy, cmfs, illuminant)
print(XYZ)

Lab = colour.XYZ_to_Lab(XYZ)
print(Lab)
```

However, it is worth to say that given the non-standardized illuminant used during the experiments (the Lumileds LED), it was not possible to compare the obtained values with any source, so it could not be proven that the tristimulus values obtained are reliable. According to the CIE parameters.

To visualize how the data was plotted when running the code, in Figure 9 it can be observed how one of the first tests, without data normalization nor applying any kind of filters to eliminate the dark noise, were showing the spectrum of the white patch of ColorChecker®.

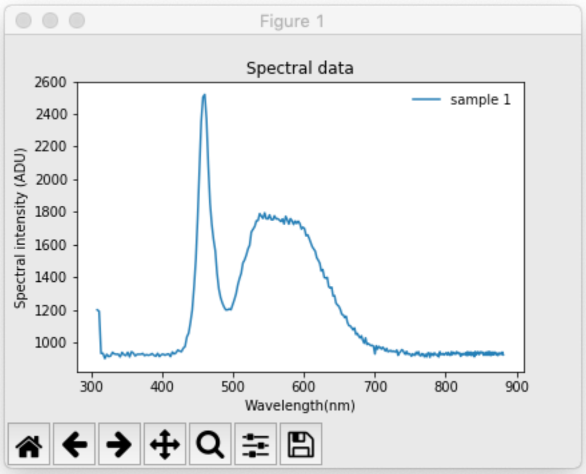

Figure 9.- Spectrum of the White Patch of the ColorChecker® when being illuminated by a D65 illuminant. No filters applied to reduce the dark noise while using a 12-bits ADC.

5.3. Dark noise reduction

Several tests were performed to determine the best set-up to produce the lower dark noise at the output of the system. Each test was repeated five times to make later comparisons.

Three different features were tested to reduce dark noise:

1) The ADC resolution, with higher values equal to lower dark noise, where four different resolutions were tested: the 8-bits ADC from Arduino Nano, the 10-bits ADC from Arduino Mega 2560, the 12-bits ADC from ESP32-DevKitC, and the external ADC of 16 bits.

2) The PGA gain value, with larger gain values equal to lower dark noise.

3) The exposure time of the sensor with respect to the light reflected by the analysed area when illuminated by an illuminant, which was proven to be significantly important to reduce the dark noise at the system's output (Joula, 2017).

5.3.1. Analog to Digital Converter (ADC)

The use of an ADC is mandatory for the acquisition of the light intensity data, as the light measurement is performed by reading the output voltage variations in a spectrometer.

An analog to digital converter (ADC) is a one-way data converter from analog signals into digital. There are many several principles for converting analog data to digital data, each offering advantages and disadvantages (Bashir et al., 2016):

1) The flash ADCs, that offers the highest sampling rate at the expense of a lower resolution.
2) The counter ramp and tracking type ADCs, which are only used in specific fields, such as CZT-based PET imaging.
3) The single and dual slope ADCs, which offer the most accurate, low power, and high-resolution data converter, being the conversion time the only limitation.

4) The SAR ADCs, well-known for being the ADC with the lowest power consumption.

5) The sigma delta (ΔΣ) ADCs, with high resolution, stability, and low power consumption.

6) The pipelined ADCs, with high speed and resolution but high conversion time.

7) The interleaving ADCs, showing a really complex architecture.

The performance comparison between the ADCs listed above can be observed in the following Table 2:

Topology	Flash	Counter	Single Slope	Dual Slope	SAR	**ΔΣ**	Pipelined	Interleaving
Sampling Rate (Samples/sec)	High (1G-10G)	Low (1-1k)	Low (100-1k)	Low (100-1k)	Medium (100K-10M)	Low (10k-1M)	Med-High (10M-100M)	Med-High (100M-1G)
Resolution (Bits)	Low (6-8)	Medium (10-12)	Medium-High (12-18)	Medium-High (12-18)	Medium-High (12-18)	High (16-24)	Medium-High (12-18)	Medium-High (12-18)
Power Consumption	High	Medium-High	Low	Low	Ultralow	Low	High	High
Latency	Low	Medium	Low-Med	Low-Med	Low	High	High	Low
Accuracy	Low	Medium-High	Variable	High	Med-High	High	Med-High	Medium
Cost	High	Low	Medium	Medium	Low-High	Low	High	High
Conversion time (No. of cycles)	1	Depends on amplitude	2^n	$2*2^n$	Variable	High	2n/2 -1	Variable

Table 2.- Performance comparison of various analog-to-digital converters (Bashir et al., 2016)

As for this work, the ADS1115 (Figure 10) was the selected ADC, a sigma delta ADC type that offers high resolution, accuracy, and stability. These three characteristics, as well as the fact that the ADS1115 has an integrated voltage reference that allows avoid voltage reference errors such as initial accuracy, long-term drift, and line and load regulation (Horowitz & Hill, 2015, pp. 683-684), makes it suitable for the system's goals.

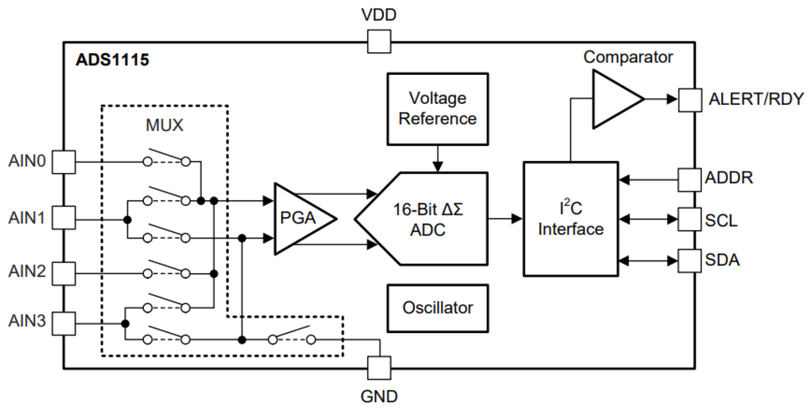

Figure 10.- ADS1115 Block Diagram Copyright® 2016, Texas Instruments Incorporated (Texas Instruments, 2018)

To test the influence and performance of the ADC conversion on the results, four different configurations were defined: one using the internal ADC of each of the three MCU tested (the Arduino Nano, the Arduino Mega 2560 and the ESP32-DevKitC), and a fourth one using the external ADC ADS1115. All four configurations are explained, and at the end of the subsection the results are shown to visualize how each configuration influence the reliability and quality of the data obtained.

The Arduino Nano is an ATmega328 based MCU that offers an 8-bits internal ADC. However, as explained in section 5.1. System's Requirements, the minimum required ADC resolution to obtain reliable data from the C12880MA spectrophotometer is 16 bits. Thus, it is expected that the result would lead to a non-reliable output with a lot of Dark Noise. Nevertheless, the configuration was tested as the Arduino Nano has been widely used in literature jointly with the C12880MA (Das et al., 2016; Laganovska et al., 2020). Almost the same configuration was used for the Arduino Mega 2560 that offers a 10-bits internal ADC. Figure 11 shows the hardware configuration with the Arduino Nano, being this schematic almost identical for the Mega 2560. For both Arduino-developed MCUs, the four channels of the sensor (video, trigger, start, clock) were connected to the analog pins A0-A3.

The hardware configuration was completed by using two additional pins for the I^{2}C communication interface of the Time-of-Flight sensor (pins A4 and A5 for the Nano, and pins D21 and D22 for the Mega 2560). The D65 Illuminant – the Luxeon 3535L LED, was connected to the digital pin D12, and the two status LEDs, that would allow the user to identify when the distance between the device and the surface to analyze was optimal, were connected to pins D11 and D10.

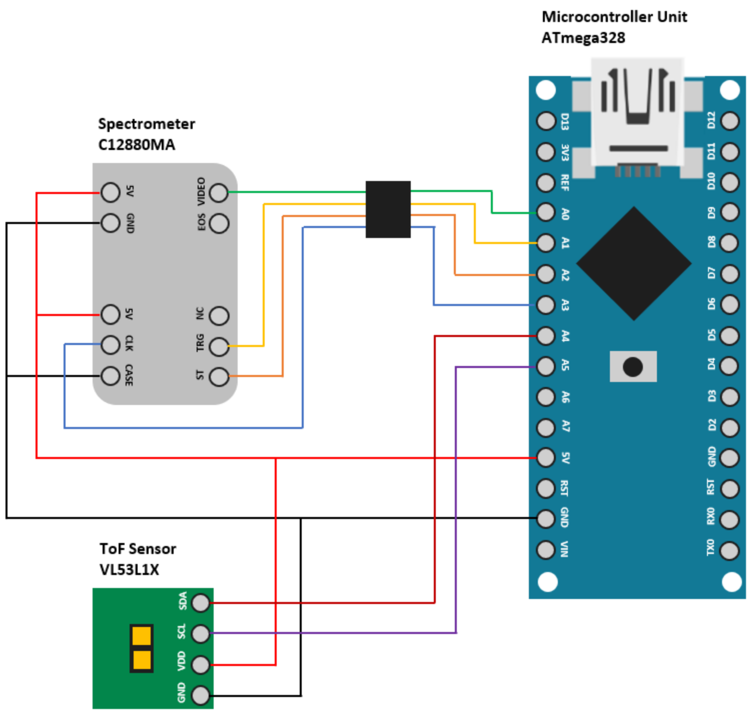

Figure 11.- Hardware schematics of the Arduino Nano prototype (LEDs not included)

As for the ESP32-DevKitC, the hardware configuration was slightly different as this MCU is not based on an ATmega Microchip that characterize the Arduino-based MCUs. The selected ESP32 has a 12-bits internal ADC, which comply with the minimum resolution required to obtain reliable data from the C12880MA. However, during the first tests, it has been observed that the results of the ESP32 ADC were highly affected by noise. Therefore, it was required to calibrate the internal ADC of the ESP32 with a lookup table to correct the ADC output. This method and its relevant script were recently developed by Weber (2019) and are freely available in the open-source repository Github. Once applied the lookup table method to correct the ADC output, the configuration shown in Figure 12 was tested.

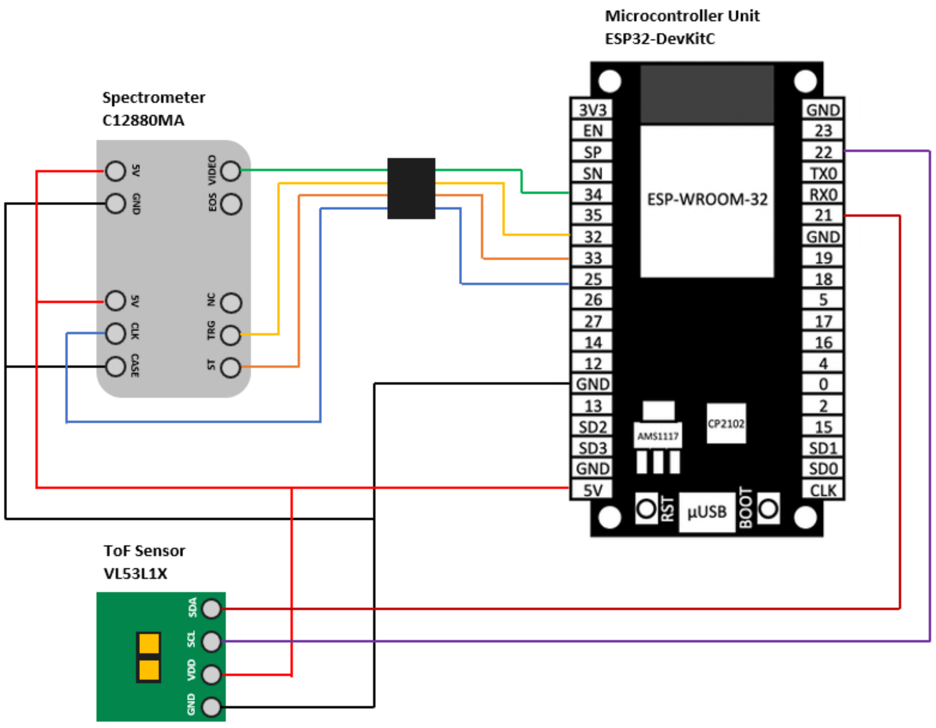

Figure 12.- Hardware schematics of the ESP32-DevKitC (LEDs not included)

The four channels of the C12880MA sensor were connected to pins 34 (only input), 32, 33, and 25 of the ESP32. It is worth to say that in the ESP32 is important to correctly identify which pins, such as the pin 34, are only for input signals and will not work as outputs. Pins 22 and 21 were used for the I2C communication with the Time-of-Flight sensor. The illuminant D65 was connected to pin 14, and the two status LEDs were connected to pins 12 and 13.

Once the three above-mentioned configurations were tested, the external ADC ADS1115 was connected as shown in Figure 13, with the gain of the PGA set to 1, equivalent to the first variable tested.

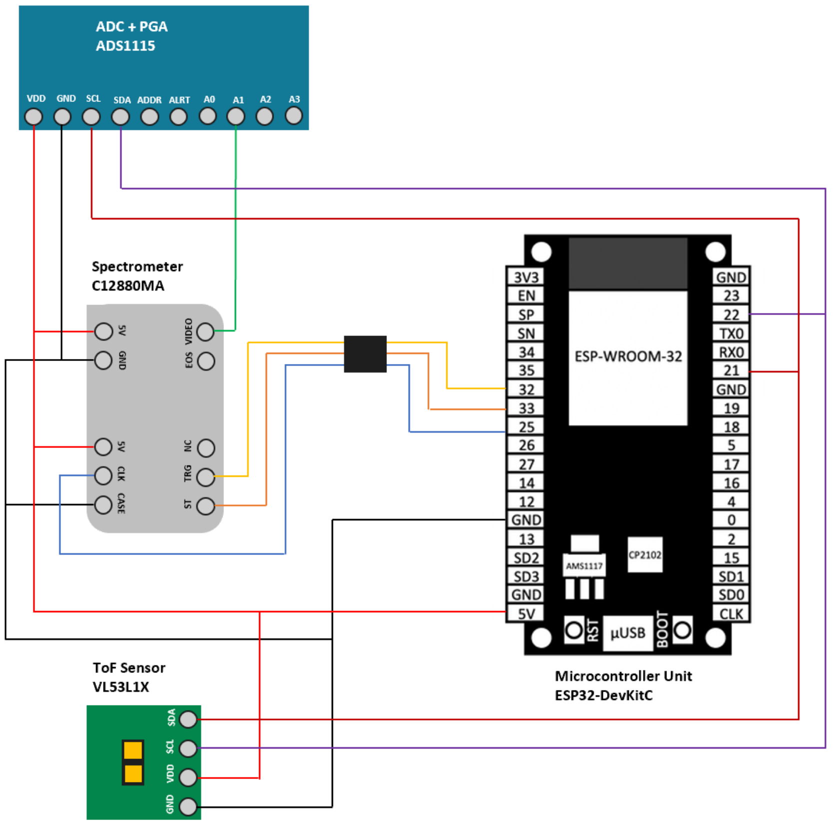

Figure 13.-Hardware schematics of the ESP32-DevKitC with the external ADC ADS1115 (LEDs not included)

The main difference between the configurations shown in Figure 12 and Figure 13 is that the video signal was connected to the external ADC's A1 pin instead of pin 34 of the ESP32. The I2C communication was used to send the digital values of the video signal to the MCU after being converted thorough the ADS1115. To allow this, it was required to do some modifications in the script that runs in the MCU, as depicted, and commented in subsection 5.2.1. Software running on the Microcontroller Unit (MCU): the definition of the pin used for the C12880MA video channel, #define SPEC_VIDEO A4, was commented in script 2; while the commented code line inside the first for loop, SPEC_VIDEO = ads.readADC_SingleEnded(1), from script 6, was uncommented.

The results obtained from testing the different hardware configurations are shown in Figure 14. All the data was normalized to allow comparisons in one plot.

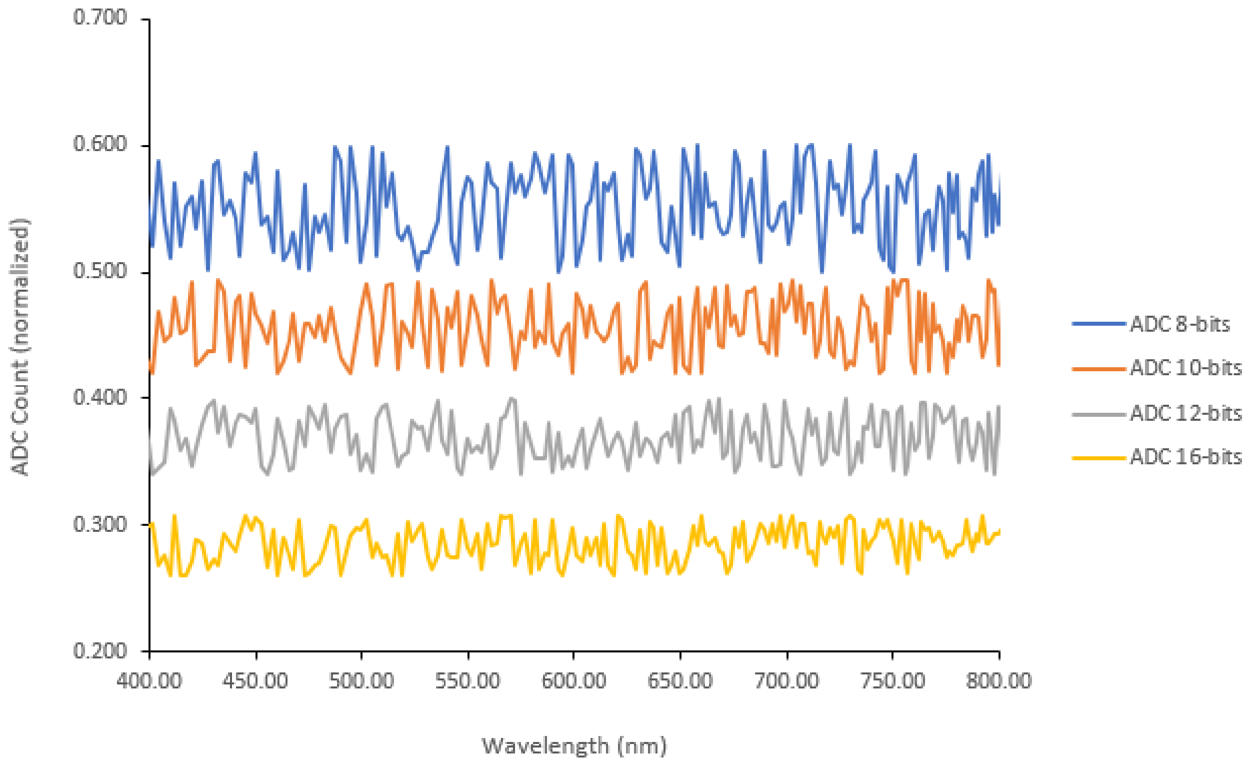

Figure 14.- Dark Noise comparison when using different MCUs with different ADCs, and using an external ADC ADS1115 with its PGA's Gain set to 1 (data processed with excel 2016)

The results show that, as expected, the higher the resolution of the ADC, the greater the quality of the results. Consequently, it was considered to use the external 16-bits ADC as shown in Figure 13 (yellow line in the plot in Figure 14) as it offers the best results among all four configurations. The reasons why the ESP32-DevKitC MCU was selected instead of the two Arduinos are that it already integrates a Bluetooth and Wi-Fi module in a small size board, and that its data rate transmission speed is considerably higher than the other two options. Moreover, there is the option to future upgrading of the prototype by adding a micro camera, as several ESP32 board versions already integrate a micro camera without requiring any extra coding for its use. The future implementation of spectrophotometric devices based on the C12880MA sensor, and the ESP32-based boards will allow easy replication and customizations according to users' analytical needs.

5.3.2. Programmable Gain Amplifier

As already mentioned in section 5.1. System's requirement, when there is a high current consumption at the video circuit of the sensor its temperature increases, causing higher dark noise. Moreover, as explained in the Hamamatsu's C12880MA spectrometer datasheet, to reduce the current flow at the video output terminal of the sensor, the analog output must be connected to a buffer amplifier before being converted to a digital signal.

In order to test how different gain values from the PGA can affect the quality of the signal obtained by the sensor when using the selected configuration, the *void setup* from script 4 (subsection 5.2.1.) was modified to enable different gain values by simply uncommenting the desired gain values and commenting the other ones. Tests were performed for gain values of two-thirds, one, two, and four to prove that the lower the gain, the less the dark.

Figure 15 shows the dark noise that affect the signal acquired at different gain values of the PGA, clearly demonstrating that dark noise increases at higher gain. According to this test, a gain of two-thirds was selected as the option that allows to the lowest amount of dark noise.

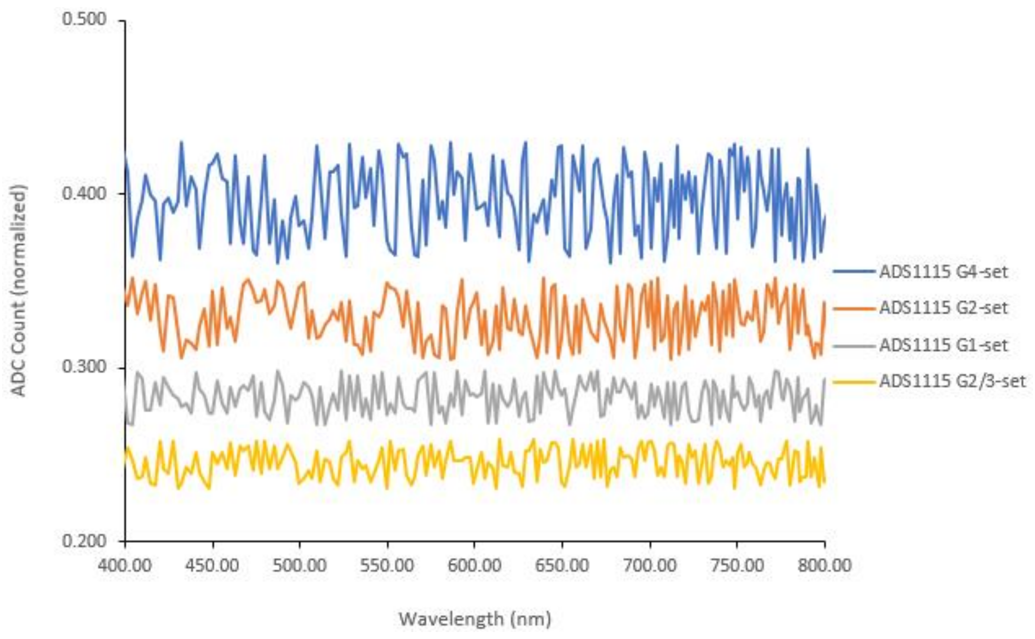

Figure 15.- Dark Noise comparison when using different MCUs with different PGA gain values (data processed with excel 2016)

5.3.3. Exposure Time

The exposure time refers to the total time that the sensor is receiving the reflected light from the illuminant on the surface to analyze. This exposure time is equal to the total integration time of the sensor, which cannot be lower than 10.8 microseconds, as commented in the Hamamatsu C12880MA spectrometer datasheet.

Giving the assumption that the longer the sensor is exposed, the more dark noise will appear at the system's output, several tests were performed by applying three different integration times: 10.8, 100, and 500 microseconds. To perform tests at different integration time values, code line, icc = spec.set_integration_time(100e-3) need to be modified (script 10, subsection 5.2.1.). As previously defined by Joula (2017), and as shown in Figure 16, the dark noise produced at the output of the system can be lowered considerably by reducing the total exposure time of the sensor to the minimum possible of 10.8 microseconds.

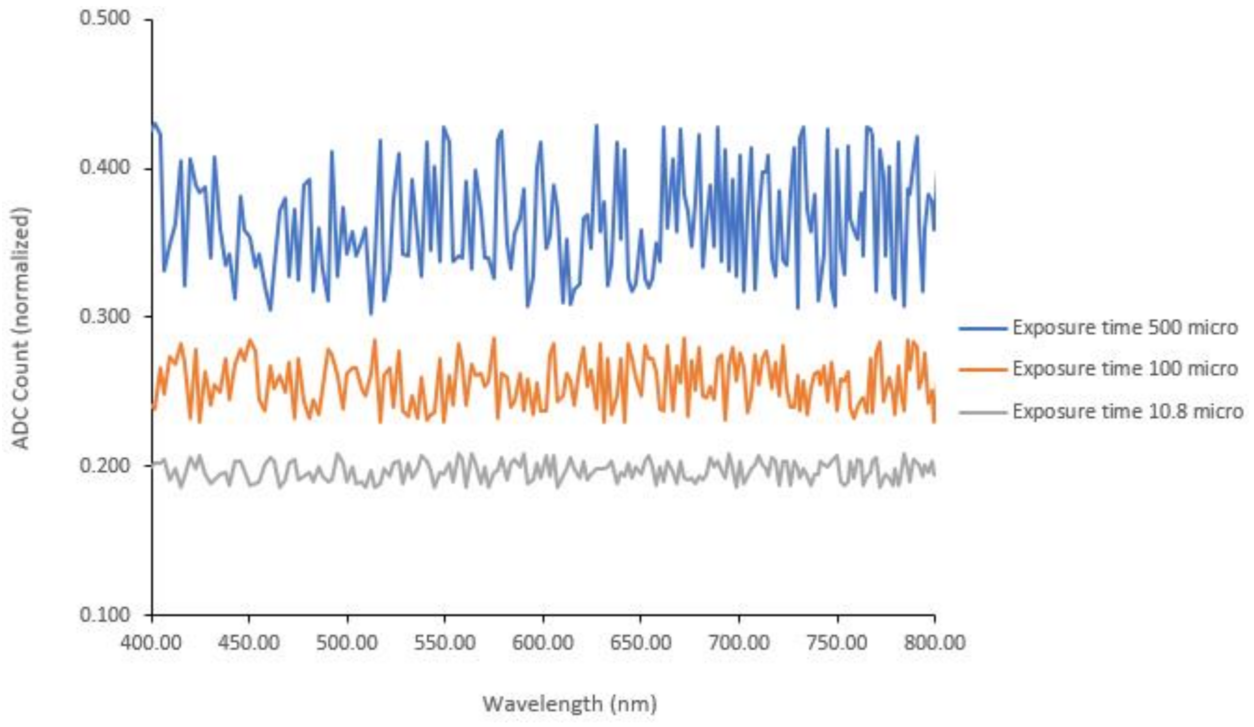

Figure 16.- Dark Noise comparison when using different MCUs with different Exposure times (data processed with excel 2016)

5.3.4. Software Smoothing Function

Once the dark noise was lowered as much as possible by modifying the hardware of the system, a software smoothing function was used to further reduce the dark noise, paying attention to not affecting the reliability of the signal obtained. It is worth to say that if this function was applied before setting the hardware as described above, the smoothing process would have been too invasive, up to modifying too much the signal acquired and generating synthetic and less reliable data. To apply the smoothing function, the software run on the CPU was modified and extra lines of code were added in script 13 (subsection 5.2.1.), as follows:

```
n = 5                     # the larger n is, the smoother curve will be        13
b = [1.0 / n] * n
a = 1
yy = lfilter(b,a,y1)

yylist = yy.tolist()
yy2 = yylist[12:]

xlist = x-axis.tolist()
y = yy2.tolist()

ylist = np.reshape(y, (-1, 1))

scaler = MinMaxScaler()
scaler.fit(ylist)
normalized = scaler.transform(ylist)
normalized1d = normalized.flatten()

plt.figure(2)
plt.plot(xlist, normalized1d)

CSVfile = np.stack([newxlist,normalized1d])
np.savetxt(file.csv', CSVfile, delimiter = ',', header = "Wavelenght(nm),Spectral
intensity (ADU)")
```

6. Designed system and results validation

In this section, the selected system is explained in detail. Moreover, tests on the red, green, and blue patches of the ColorChecker® were performed to show the performance of the system.

6.1. System design

The tests performed and commented in the previous section 5. Hardware and Software Development and Tests, allowed to select the most appropriate design for the system. Among the three MCUs that were tested, the ESP32-DevKitC was selected for three main reasons:

1) It allows to develop a system that could be easily upgraded in the future by taking advantage of the considerable number of analog and digital pins available.
2) Numbers of available pins being approximately equal, the ESP32-boards show reduced dimensions if compared to the Arduino Mega 2560, which allows to implement highly portable devices.
3) The ESP32-DevKitC already integrates a wireless connection module, so avoiding adding extra components that would have required to change from serial communication to a wireless one.

As for the ADC, it was decided to select the one with the highest possible resolution, as one of the project aims is to develop a system a capable to acquire reliable signals. The ADC selected is the ADS1115 as it offers a resolution of 16 bits and has an internal PGA, thus solving two problems in one single component. After testing the PGA, the x2/3 gain was selected to lower the dark noise at the output of the system.

As for the software, the script was modified in order to determine an optimal exposure time of 10.8 microseconds. Finally, a smoothing function was applied to eliminate the residual dark noise without reducing the robustness of the signal acquired.

All these steps and consequent decisions have led to the design of the system shown in Figure 17. This includes three LEDs - the illuminant D65 aka the Lumileds Luxeon 3535L and two status LEDs, and a microSD card reader to store all the acquired and processed information.

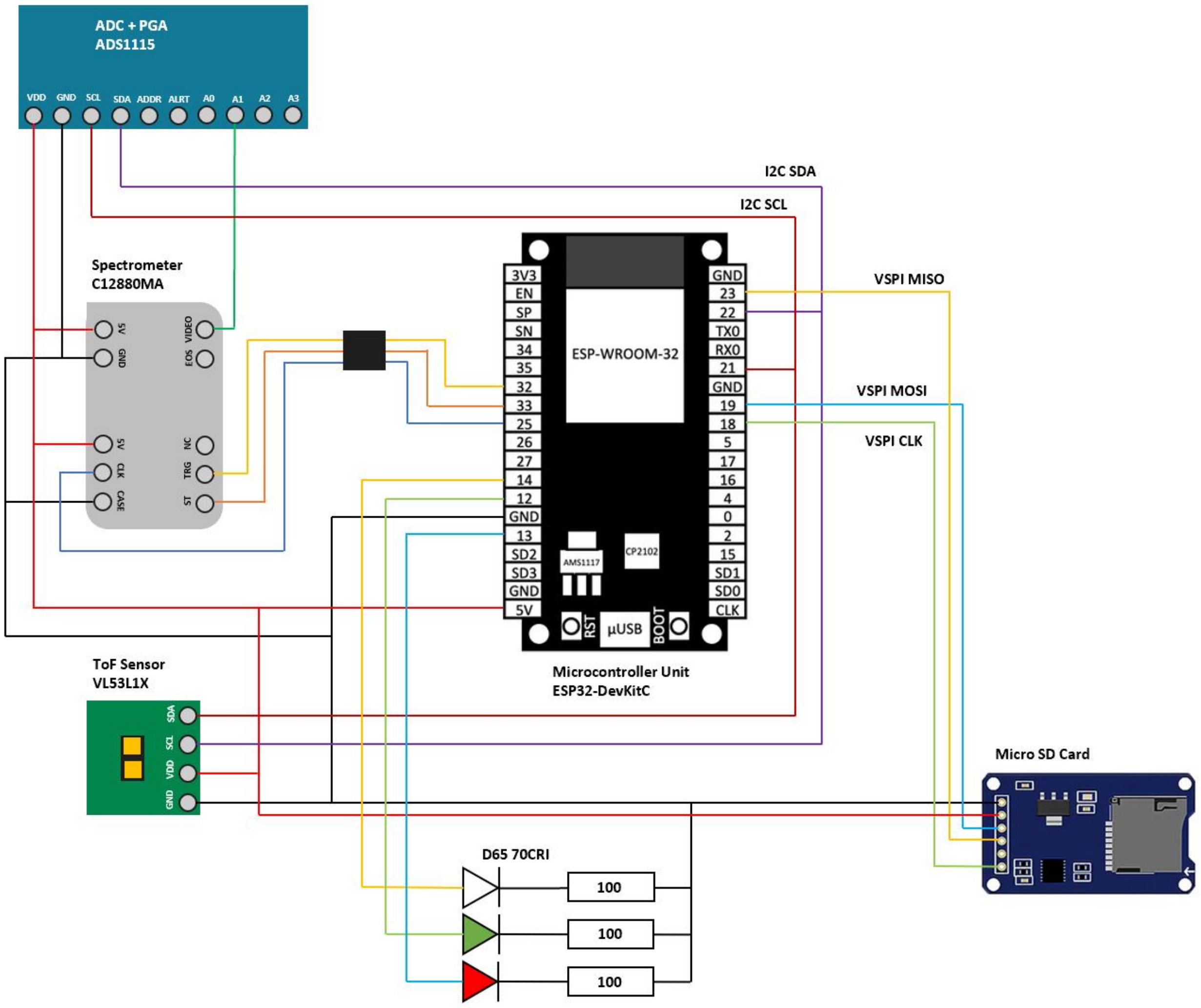

Figure 17.- Hardware schematics of the final device, including all hardware components used.

6.2. Results on red, green, and blue patches

Once the system was built, it was tested again using the red, green, and blue patches of the ColorChecker® to visualize how the system works when analyzing colors others than white and black. No spectral nor CIE-defined tristimulus values information are made available by the Lumileds Luxeon 3535L producer. This makes impossible to perform comparative analysis by applying standard deviation formulas such as the deltaE defined by the CIE. Therefore, comparisons need to be limited to visual comparison between the spectral plots. Notwithstanding, the tests on the red, green, and blue patches were performed to validate the sensibility of the C12880MA sensor to the subtractive primary colors Red (R), Green (G) and Blue (B). The plots of the spectra of these colors are shown in the following Figure 18 with visible colors in background to give a visual perception of the results:

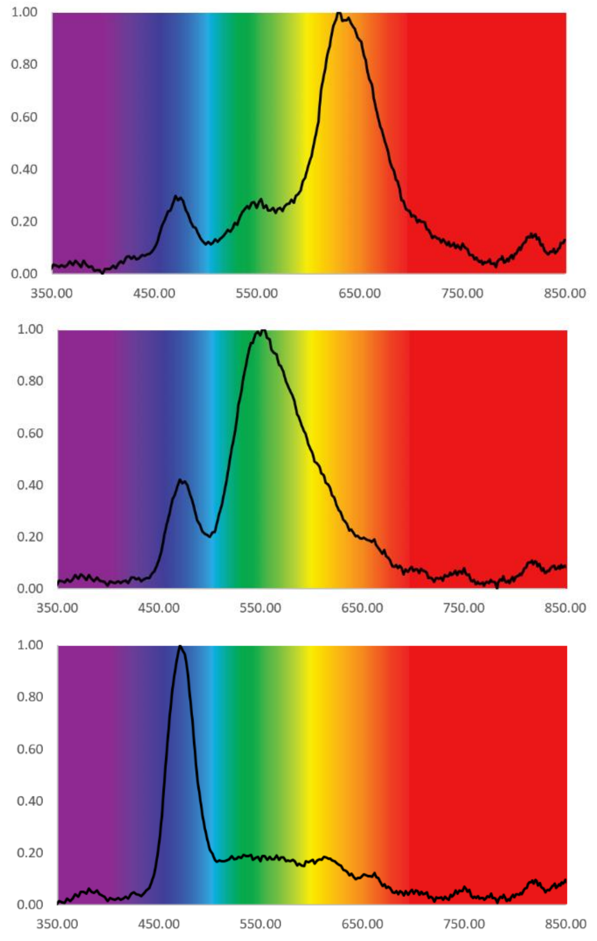

Figure 18.- Spectrum of the red, green, and blue patches of ColorChecker® when illuminated by a D65 70CRI illuminant at the distance of 5mm. (red on top, green on the middle, and blue at the bottom)

By examining these plots, it can be observed that the reflectance spectra of R, G and B colours are consistent with the wavelenght range of these primary colours: 620-750 nm, 495-570 nm, and 450-475 nm, respectively. However, the dominant wavelenght of the blue and green patches (470.33 nm, and 554.09 nm respectively) are quite compatible to those obtainable by illuminating the samples with a CIE standardized D65 illuminant. Conversely, the dominant wavelenght of the red patch (636.87nm) shifted towards the orange-yellowish band of the spectrum. Notwithstanding, this mismatch cannot be ascribed to a less sensitivity of the C12880MA sensor to the red band of the spectrum, but to the tendency of the Lumileds Luxeon 3535L LED to emit a white light that tends towards the yellowish band of the spectrum (Lumileds, 2021)

7. Conclusions

In this work, the characteristics of the spectrophotometric sensor C12880MA were explored in deep to facilitate its integration into an open-hardware system and optimize its data acquisition and processing flow. As observed in other open-hardware prototypes developed during the last few years and based on the use of the same C12880MA, the main drawbacks were the low resolution and the dark noise, both factors highly limiting the ability of the sensor to acquire

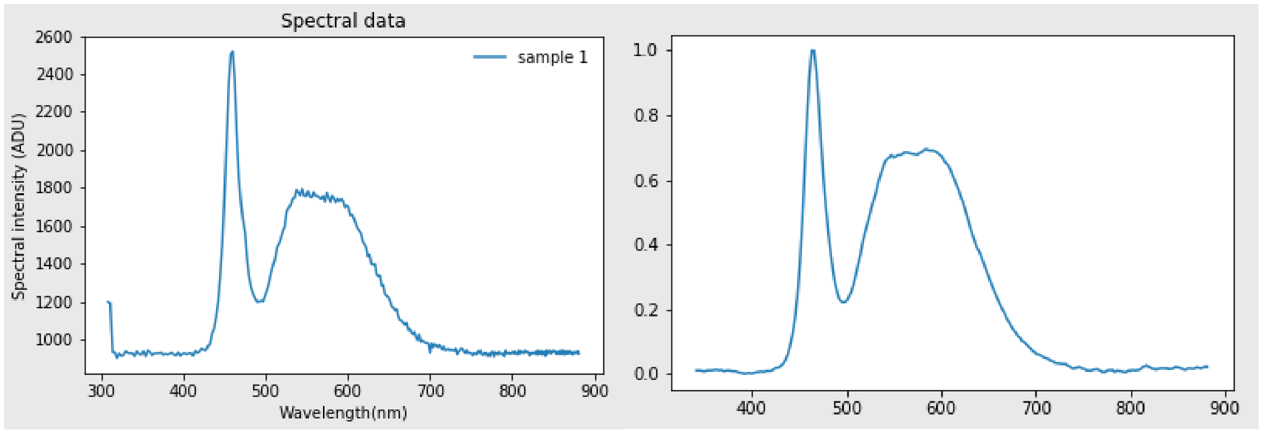

Figure 19.- On the left, D65 70CRI spectrum while using the standard hardware configuration presented by various authors. On the right, the improved spectrum after eliminating dark noise.

accurate and precise data. Thanks to the hardware and software solutions adopted, the resolution was increased up to the maximum level to allow a reliable acquisition of the electromagnetic signal, and the dark noise at the output of the system was lowered enough to increase the quality of the data acquired by the sensor. Figure 19 shows the spectral responses of the sensor before and after optimizations. Specifically, these optimizations were achieved by means of:

- The selection of a correct and powerful ADC
- The correct set-up of the op-amp gain value
- The adjustment of the exposure time of the sensor
- The use of a smoothing equation.

Even more, the prototype was designed specifically to pursue non-contact, non-invasive spectrophotometric analysis, thus satisfying the requirements of many applications into the domain of reflectance spectrophotometry. Indeed, the use of the distance sensor, as well as the geometry of the incident light of the integrated illuminant, allow to respect a minimum distance from the analyzed object of approximately 5 mm. This distance not only allows avoiding any contact of the head of the prototype with the surface to be analyzed, but even allows to limit the area to be analyzed, thus improving analytical precision.

Finally, it is worth to remind that the spectrophotometric prototype, notwithstanding the expected improvements listed in the next subsection, is already used within the project "BioACHS – Biodegradation Assessment for Cultural Heritage surfaces and substrates", jointly carried by the Beta Technological Center and the Mecamat research group to check the color variations in painted surfaces affected by biological patinas. As for the that reason, once this TFG will be discussed, it is expected to submit a co-authored scientific article to a scientific publication.

7.1. Future improvements

Although the very good results achieved, more efforts are expected to move from the proposed prototype to a more robust, definitive device. First, the choice of the illuminant needs to be rethought according to the new technical suggestions of the CIE, the international body that works on standards and definitions in the field of light, color, and spectrophotometry. Although the LED used emits a standard D65 white light, and its chromaticity white point values are very close to those of the theoretical LED B-5 recommended by the CIE, at that point it is not possible to perform reliable transformations of the spectra acquired into triplets of the CIE color spaces. However, it is worth to remind that the LED-based illuminants very recently defined by the CIE, such as the above-mentioned B-5, are still theoretical LED showing theoretical spectra, so that none of the open-hardware spectrophotometers developed so far is using a CIE-standardized LED. Consequently, none of the spectra captured by other open-hardware devices can be converted into CIE accepted color spaces. Second, the design of the case will be modified to housing optical filters that will allow to select the type of light to be acquired, according to specific analytical needs. Third, as based on the ESP32 MCU, the overall system still allows not mandatory but desirable hardware improvements, such as the integration of a micro-camera to help visualize the spotted areas before the data acquisition. Finally, the development of a dedicated Graphical User Interface would highly improve the usability of the final device also by a non-specialized end-user.

8. References

[1] Hamamatsu. (2021) *C12880MA Datasheet, “Finger-Tip Sized, Ultra-Compact Spectrometer Head Supporting High Sensitivity and Long Wavelength Region”.* [online] https://www.hamamatsu.com/resources/pdf/ssd/c12880ma_kacc1226e.pdf (Last accessed on 2021-05-03)

[2] Arduino. (n.d.) *Arduino Products.* [online] https://www.arduino.cc/en/Main/Products (Last accessed on 2021-04-25)

[3] Espressif. (n.d.) *ESP32 Hardware, “The Internet of Things with ESP32”.* URL: http://esp32.net/#Hardware (Last accessed on 2021-04-25)

[4] Versek, C., Woodworth, P., Meckes, A., Blair, D. (2016). *Review of the Hamamatsu C12880MA Micro spectrometer Module.* [online] https://impfs.github.io/review/ (Last accessed on 2021-05-03)

[5] Joula, J. (2017). *Prototyping electronics and software for a spectrometer module.* M.A. Thesis. Aalto University.

[6] Das, A., Wahi, A., Kothari, I., & Raskar, R. (2016*). Ultra-portable, wireless smartphone spectrometer for rapid, non-destructive testing of fruit ripeness*. Scientific Reports, 6(1), 32504–32504. https://doi.org/10.1038/srep32504

[7] Laganovska, K., Zolotarjovs, A., Vázquez, M., Mc Donnell, K., Liepins, J., Ben-Yoav, H., Karitans, V., & Smits, K. (2020). *Portable low-cost open-source wireless spectrophotometer for fast and reliable measurements.* HardwareX, 7, e00108–. https://doi.org/10.1016/j.ohx.2020.e00108

[8] Pérez, F. Granger, B. (2012). *An open-source framework for interactive, collaborative, and reproducible scientific computing and education*.

[9] Kim, Y., Seung-Taek, O., & Lim, J. (2018). *Development of Portable Spectrometer supporting, Automatic Control of Integration Time.* Research Journal of Pharmacy and Technology, 11(10), 4619–4626. https://doi.org/10.5958/0974-360X.2018.00845.4

[10] Sandak, J., Sandak, A., Zitek, A., Hintestoisser, B., & Picchi, G. (2020*). Development of Low-Cost Portable Spectrometers for Detection of Wood Defects.* Sensors (Basel, Switzerland), 20(2), 545–. https://doi.org/10.3390/s20020545

[11] Gaudenzi Asinelli, M., Català, P., Serra, J., Serra, M., Jerez, R., Molera J. (2019). *The Open-Source hardware paradigm: an emerging model for research?* In: ERSCP2019 CERCA REFERENCE

[12] Blum, P. (1997). *Physical properties handbook: a guide to the shipboard measurement of physical properties of deep-sea cores.* (Rev. Ed.). (pp. 75-85). Texas A&M University.

[13] Richard W Harold. (2001). *An introduction to appearance analysis.* GATFWORLD (Pittsburgh, Pa.), 13(3), 5–.

[14] Lumileds. (n.d.). *Luxeon 3535L Line datasheet, "High efficacy in a 3535 package with full range of CCTs and CRIs".* [online] https://www.lumileds.com/wp-content/uploads/files/DS203-LUXEON-3535L-Line-datasheet.pdf (Last accessed on 2021-05-03)

[15] Weber, H. (2019). *ESP32: How to correct the ADC*. Github. [online] https://github.com/MacLeod-D/ESP32-ADC (Last accessed on 2021-05-06)

[16] Wang, Z., Zhao, B., Li, J., Luo, M., Pointer, M., Melgosa, M., & Li, C. (2017). Interpolation, extrapolation, and truncation in computations of CIE tristimulus values. Color Research and Application, 42(1), 10–18. https://doi.org/10.1002/col.22016

[17] Pearce, J. (2017a). *Impacts of open-source hardware in science and engineering*. The Bridge (Washington, D.C.: 1969).

[18] Moritz, M., Redlich, T., Günyar, S., Winter, L., & Wulfsberg, J. (2019). *On the Economic Value of Open Source Hardware – Case Study of an Open Source Magnetic Resonance Imaging Scanner*. Journal of Open Hardware, 3(1), 1-9. https://doi.org/10.5334/joh.14

[19] Pearce, J. (2017b). *Emerging Business Models for Open Source Hardware*. Journal of Open Hardware, 1(1), 1-14. https://doi.org/10.5334/joh.4

[20] Pearce, J. (2020). *Economic savings for scientific free and open source technology: A review*. HardwareX, 8, e00139–e00139. https://doi.org/10.1016/j.ohx.2020.e00139

[21] Ibsen Photonics. (n.d.). *Signal-to-noise ratio and dynamic range definitions*. [online] https://ibsen.com/technology/detector-tutorial/signal-to-noise-ratio-and-dynamic-range-definitions (Last accessed 25-05-2021)

[22] Technote. (2011). *Units and conversions for dynamic range and noise values*. [online] https://image-engineering.de/library/technotes/748-units-and-conversions-for-dynamic-range-and-noise-values (Last accessed 25-05-2021)

[23] Bashir, S., Ali, S., Ahmed S., Kakkar V. (2016). *Analog-to-digital converters: A comparative study and performance analysis*. 2016 IEEE International Conference on Computing, Communication and Automation, 1-7. https://doi.org/10.1109/CCAA.2016.7813861

[24] Texas Instruments. (2018.) *ADS1115 Datasheet*, *"ADS111x Ultra-Small, Low-Power, I^{2}C-Compatible, 860-SPS, 16-Bit ADCs With Internal Reference, Oscillator, and PGA".* [online] https://www.ti.com/lit/ds/symlink/ads1113.pdf?ts=1621951082705&ref_url=https%253A%252F%252Fwww.google.com%252F (Last accessed 25-05-2021)

[25] Horowitz, P., & Hill, W. (2015). *The Art of Electronics* (3. ed). Cambridge University Press. 1220pp.

[26] Li, Z., Seering, W., (2019). *Does Open Source Hardware Have a Sustainable Business Model? An Analysis of Value Creation and Capture Mechanisms in Open Source Hardware Companies*. In Proceedings of the 22nd International Conference on Engineering Design (ICED19), Delft, The Netherlands, 5-8 August 2019, 2239-2248.

[27] Hoshi, T., Ueda, K., Takikawa, Y., Azuma, T., (2018). *Digitally Fabricated Mobile Spectrometer for Multipoint Continuous Spectroscopic analysis of Light Environment in Greenhouse Tomato Canopies*. Environmental Control Biology 56(4), pp. 149-155. https://doi.org/10.2525/ecb.56.149

[28] Sosa-Herrera, J.A., Vallejo-Pérez, M.R., Álvarez-Jarquín, N., Cid-García, N.M., López-Araujo, D.J., 2019. *Geographic object-based analysis of airborne multispectral images for health assessment of Capsicum annuum L.* crops. Sensors 19(4817), pp. 1-21 https://doi.org/10.3390/s19214817

[29] CIE Publication No. 167:2005. Recommended practice for tabulating spectral data for use in colour, Central Bureau of the Commission Internationale d'Eclairage (2005)

[30] Westland, S. (2015) Interpolation of Spectral Data. In: Luo R. (eds) Encyclopedia of Color Science and Technology. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-27851-8_366-1

[31] Català, P., Gaudenzi Asinelli, M., Martinez, S., Muñoz, A., Pomés, J., Rovira, A., (2017). *Demonstration of mechatronics solutions for on site and lab based challenges. 7th International Workshop on Higher Education*, 26 June 2017, Vic (Spain). Oral communication. [online] https://mon.uvic.cat/international-workshop/mechatronics-robotics-materials-and-signal-processing/ (Last accessed 17-05-2021)

[32] GroupGets. (2016). *C12880MA Breakout Board v2 by GetLab*. [online] https://groupgets.com/manufacturers/getlab/products/c12880ma-breakout-board-v2 and https://github.com/groupgets/c12880ma (Last accessed 25-05-2021)

[33] Malacara, D. (2011) *Vision and Colorimetry: Theory and Applications*. (2. ed). SPIE Press https://doi.org/10.1117/3.881172

[34] McCluney, R. (2014*). Introduction to radiometry and photometry*. Artech House.

[35] Schubert, F. (2006). *Light-Emitting Diodes.* (2. ed). Cambridge University Press https://doi.org/10.1017/CBO9780511790546

[36] Color Matters. (n.d.) *Look Inside the Eye*. [online] https://www.colormatters.com/color-and-vision/look-inside-the-eye (Last accessed 26-05-2021)

[37] Poynton, C. (2012). *Digital video and HD: Algorithms and Interfaces*. (2. ed). Morgan Kaufman Series in Computer Graphics. ISBN-10: 9780123919267

[38] Whetzel, N. (2016). *CIE Standard Observers and calculation of CIE X, Y, Z color values.* [online] https://support.hunterlab.com/hc/en-us/articles/203420099-CIE-Standard-Observers-and-calculation-of-CIE-X-Y-Z-color-values-AN-1002b (Last accessed 26-05-2021)

[39] Nickerson, D. (1960). *Light sources and color rendering*. JOSA 50 (1): 57-69 https://doi.org/10.1364/JOSA.50.000057

[40] CIE. (2004). *CIE Colorimetric and Colour Rendering Tables*, Disk D002, Rel 1.3 [online]

[41] CIE. (2018). CIE 15:2018 Technical Report: Colorimetry (3. ed.). 10 CFR 430 Subpart B, App. R, 4.1.1. International Commission on Illumination.

[43] Lumileds. (2021). Luxeon 3535L Line: “High efficacy in a 3535 package with full range of CCTs and CRIs”. [online] https://www.lumileds.com/wp-content/uploads/files/DS203-LUXEON-3535L-Line-datasheet.pdf (Last accessed on 2021-06-06)

[44] Wikiwand. (n.d.). Standard Illuminant: “White points of standard illuminants”. [online] https://www.wikiwand.com/en/Standard_illuminant

[45] STMicroelectronics. (2021). VL53L1x datasheet: “A new generation, long distance ranging Time-of-Flight sensor based on ST’s FlightSense^{TM} technology”. [online] https://www.alldatasheet.com/view.jsp?Searchword=Vl53l1x%20datasheet