NeoPixel Sunrise Clock

An intelligent bedside clock

Presented by:
Othniel Konan
KNNOTH001
Dept. of Electrical and Electronics Engineering
University of Cape Town

Prepared for:
Dr. Simon Winberg & Mr. Justin Pead
Dept. of Electrical and Electronics Engineering
University of Cape Town

Submitted to the Department of Electrical Engineering at the University of Cape Town in partial fulfilment of the academic requirements for a Bachelor of Science degree in Electrical and Computer Engineering

November 11, 2017

Key Words: Neopixels, Circadian rhythm, STM32, Blue light, Nextion, MIT App Inventor

Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this report from the work(s) of other people has been attributed, and has been cited and referenced.
3. This report is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as their own work or part thereof.

Signature: .....
Othniel Konan

Date: .....

Acknowledgments

Abstract

Studies made on the human behavioural patterns revealed the existence of an endogenous clocks controlling the human circadian rhythm. Among these clock, the master clock located in the suprachiasmatic nuclei responsible for the control of the human sleep-wake cycle is influenced by light.
This project aims to create a device capable of emitting light which has having both soporific and alarming effects on human beings to control the human sleep-wake cycle. The device is expected to be an alarm clock which can be used as a supplement in the cure of sleep disorder.
The desgin has three focus areas: the hardware, software and mechanical design. The hardware design consists of the identification and selection of components required to make the device an alarm clock that can meet the light requirements. The mechanical design is paired with the hardware design to ensure that the device can fit in a $25 * 25 * 10cm^3$ case. The software design consists of identifying the abstract modules and designing when required communication protocols between these modules. in this section, it was decided that the device will use a ring consisting of 180 neopixels as its light source, will require external storage capacity and a real time clock with an alarm functionality, will be controlled by an onboard touchcreen and a smartphone application. The system was tested using software unit tests and a logic analyser for the hardware modules. The light source illuminance, current drawn per colour and temperature rise were quantified during eperiments.
The ring of neopixels performed well above expectations; it emits light of $465 - 467nm$ wavelength with an illuminance above 30lx on any object placed at maximum of 1m at a maximum angle 90 deg to the normal of the ring.

Contents

List of Figures

List of Tables

Listings

Chapter 1

Introduction

1.1 Background to the study

Human behavioural and anatomical activities are influenced by several internal cycles. Among these internal cycles is the **circadian rhythm**, a rhythm studied for many years and whose impacts on the human activity have led to new interests in the regulation of these activities. Formally defined as a "cyclical changes in hormones, body temperature, and other biological processes over the course of a 24 hour period" [1], the Natural Institute of Health (NIH) defines it as "a physical, mental and behavioural changes that follow a roughly 24-hour cycle, responding primarily to light and darkness in an organism's environment"[1]. The circadian rhythm plays an important role as it also affects the human sleeping and rising pattern. The circadian rhythm is influenced by the production of *melatonin* produced by the *pineal gland* whose activities are dependent on the presence of light on the *retinal-hypothalamic tract*[3].These studies have shown that the presence of light of specific wavelength at certain period of time during a day can affect the normal sleeping cycle.

According to the NIH, there is a correlation between long-term health problems and sleep disorders [6]. While stress levels and lifestyles affect the sleeping pattern, there is a strong evidence that the human sleep-wake cycle is strongly affected by light. With the invention of the electric light and the recent human exposure to LED screens, humans have more exposure to nocturnal light. Recent researches have shown that the usage of LED technologies at night is linked to sleep deficiency. Blueish light is said to have a huge impact on one of the human internal clock[4]. Sleep deficiency due to inappropriate light exposure can be cured using an optimal light exposure[4]. Researchers were able to quantify, qualify and time the light that is suitable to maintain the natural sleep-wake cycles [2]. With these results, it is possible to create an environment that will follow user specific light requirements needed to treat patients having a sleep disorder.

1.2 Objectives of this study

1.2.1 Problems to be investigated

This project investigates the feasibility of making a user-friendly embedded system, relatively cheap that could be used as a personal medical device in solving human sleep disorder. This problem envelops the following question:

1 Is this possible to use a programmable light source to emit light of around $460nm$ at $30lux$?
This is the light requirement as mentioned in section 1.3.

2 Can an embedded system meeting the light requirement mentioned above be used as a personal medical device?
The question focuses on the future use of the device in regulating the human sleep-wake cycle by medical prescription of light requirement.

1.2.2 Purpose of the study

The purpose of this study is to create a device that can be used to regulate the human sleep-wake cycle while being user-friendly and a personalisable digital alarm clock. The product would need to be relatively cheap and have more features than its competitor. Moreover, the device should be able to use user-specific data in the regulation of the sleep-wake cycle.
Further objectives [1] includes:

- The use of user-specific medical lighting requirements and patterns to be used a personal medical device supplementing sleeping disorder treatment.
- Ability to pull events from an online calendar and set these events as alarms.
- User authentication for onboard screen usage and bluetooth connection

1.3 Scope and limitations

The scope of this project involves the design of an functional embedded system named NeoPixels Sunrise Clock also known as NPSC, capable of producing light of $460nm \pm 10nm$ with an intensity of $30lux$ as mentioned by the paper *"Action Spectrum for Melatonin Regulation in Humans: Evidence for a Novel Circadian Photoreceptor"*. The code and design artefact repository and a full documentation including a user manual, for anybody who wants to make use of the code design resources, also need to the delivered. Moreover, a description of future use of the device in the study of the effect

[1] These are sub-objectives that would be implemented depending on the time available

of light on the circadian rhythm will be required.
This project does not study the effect of light on the users. For ethical reasons, the NPSC will not be tested on human subjects in real situations of either waking humans or including lighting to facilitate sleep at night. Instead, the system will be tested based on the recommendation from the research literature.
The design and creation of the NPSC are subject to several constraints listed below:

- **Time:** The project has a duration of 12 weeks within which the research, design, development, implementation, verification, and report writing need to be done.
- **Budget:** The project budget allocation is **R1000**
- **Light:** The NPSC must be able to produce blue light with a wavelength of $460nm$ while providing enough light to meet the requirement of the research paper and provide a various range of colour for sunrise simulation. These requirements narrow the options for choosing the right light emitters.
- **Size:** The NPSC is meant to be a bedside lamp, this implies that it should have a relatively small size to be able to fit on a $50cm * 50cm$ bedside table. A volume of $25 * 25 * 5cm^3$ is the target volume for the final product.

1.4 Plan of development

The project was broken into sections and subsections with an estimated timeline.
The *Gantt Chart* used for this project is shown in fig. 1.1. The project started with the an intensive research on the science related to the human sleeping cycle. The research lead to the design of the NPSC consisting of its hardware and software modules. During the manufacturing process, the software framework of the NPSC was continously improved. The NPSC hardware and software integration were done later after the assembly of the hardware. Finally, the software was improved during the remaining lifetime of the project.

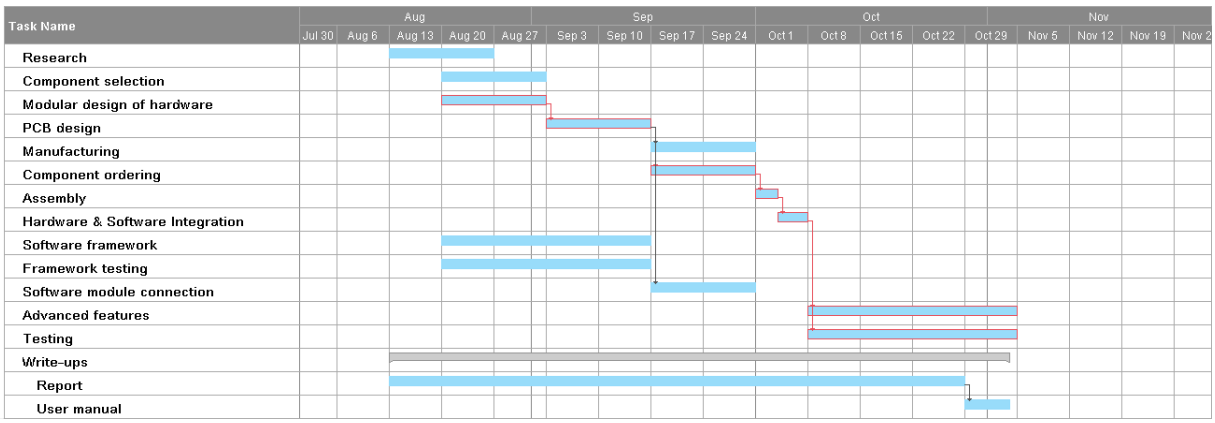


Figure 1.1: Gantt chart showing the timeline of every task in the project as well as its critical path.

1.4.1 Chronological progression of the report

The report organisation is displayed in fig. 1.2. The sections of the report are explained below:

- **Research**
  - **Introduction:** The feasability of the project as well as its scope and limitations are defined in the introduction.
  - **Literature Review:** The literature review gives an insight in the researches made for this project. This includes scientific discoveries on the human sleeping cycle, experiements and results performed by researchers on that matter, and some technical engineering design decisions.
- **Design**
  - **Methodology:** This section covers the hardware, softeare, and mechanical design of the NPSC.
  - **Results:** This section displays the results of the hardware and software testing.
- **Write-ups**
  - **Discussion:** The analysis of the results obtained. Here, the performance of the NPSC is evaluated. A costs and functional analysis of NPSC done to evaluate its performance compared to its competitors. Moreover, the future use of the NPSC is elaborated.
  - **Conclusion:** An evaluation of the project, did we achieve the intended goals.
  - **Recommendations:** We dive into the solutions or recommendations that could improve the design of such device.
  - **User manual:** This section is for any users of the NPSC. It provides a clear explanation of the features of the NPSC and a detailed manual.

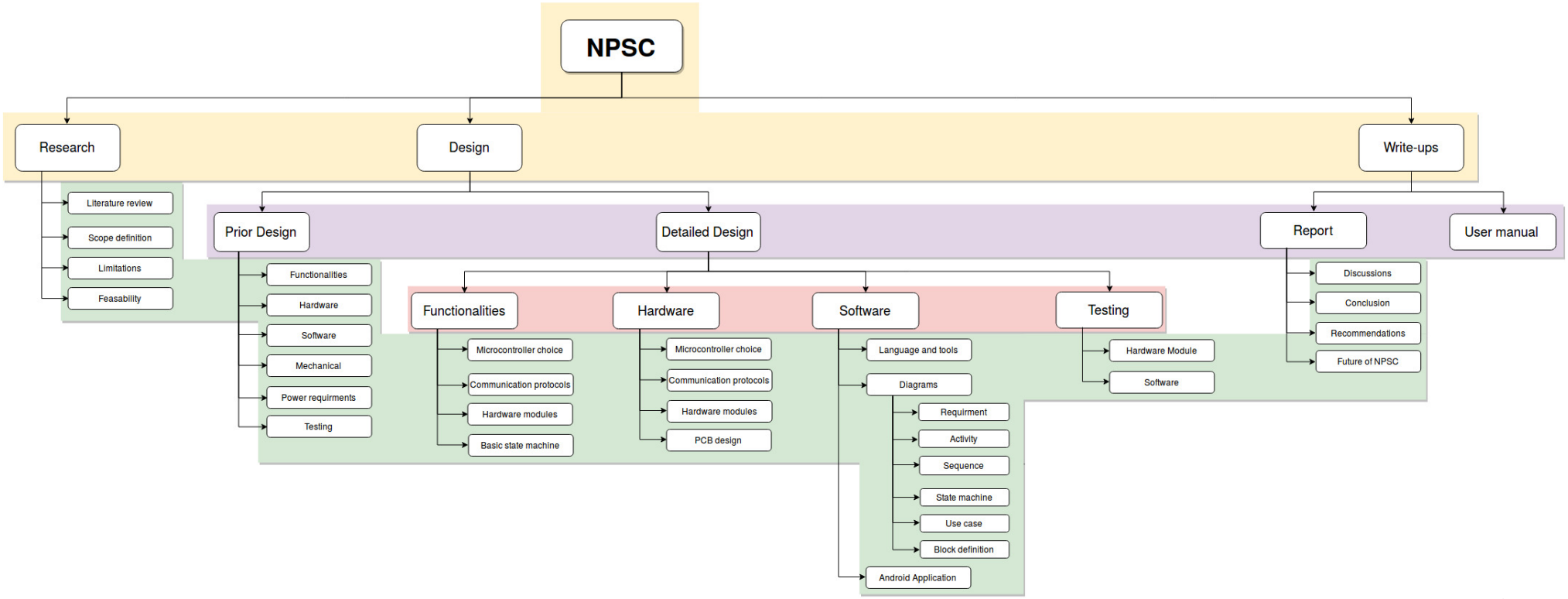


Figure 1.2: Report breakdown detailing the different sections needed to be included in the report.

Chapter 2

Literature Review

This chapter reviews the research papers, articles, books and other relevant forms of research used in the design of the NPSC. It has been divided into the sections illustrated by fig. 2.1: The literature review starts with an explanation of the problems to be solved,

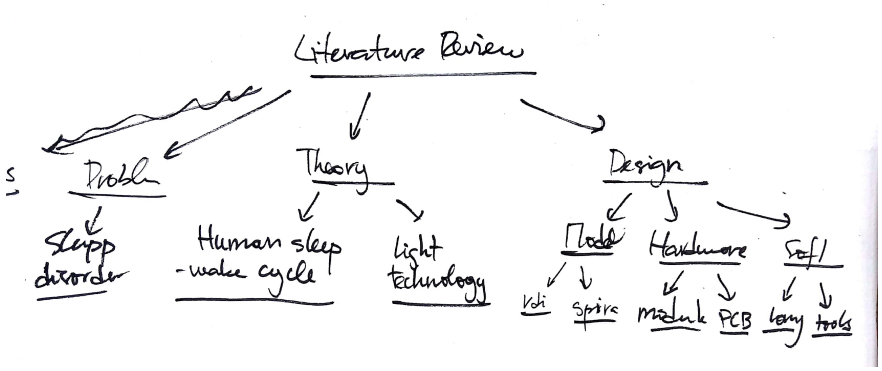


Figure 2.1: Overview and classification of the sections of the literature review.

it continues by uncovering the theory behind these problems and ends with a review of the design methods, hardware components and software tools used in this project.

2.1 The human sleep-wake cycle

Human has many "internal clocks" among them is the master clock located in the suprachiasmatic nuclei. These internal clocks or endegenous clocks are internal mechanisms in organisms reponsible for the regulations of certain functions or activities. The master clock cheif among them, regulates the secretion of melatonin is affected by light. The creation of artifical light expecially LEDs have caused a disruption in the sleep-wake cycle. This section give an overview of the human internal clocks and how they are affected by light.

2.1.1 The circadian rhythm

Human seasonal behaviours are synchronised to the environment by **biological clocks** responsible for the creation of biological rhythms. Biological clocks which are composed of proteins that act reciprocally on the body's cells are the natural timing devices found in many organs. The discovery of the genes from these biological clocks responsible for the control of these rhythms was made by three scientists Jeffrey C. Hall, Michael Rosbash and Michael W. Young winners of the Nobel Prize in Physiology or Medicine [14]. Their discovery made in the 1980s led to advanced research on the role of the circadian rhythms.

Circadian rhythms are biological rhythms which follow the same pattern in absence of external cues (endogenous), are influenced by the presence of external stimuli (entertainable), oscillating roughly every 24h[1] over a range of physiological temperatures. In the presence of external stimuli -also known as *zeitgebers*-, circadian rhythms synchronise their periodicity with these external stimuli. The zeitgebers of the circadian are the daily variation of the temperature and the dark/light cycle of the day.

Circadian rhythms have endogenous and exogenous components. Human placed in isolation without knowledge of the time continued exhibiting a circadian rhythm with their pacemakers notably the melatonin secretion, sleep-wake cycle, body temperature [13]. These results prove the existence of endogenous components of the circadian rhythms as it illustrates the effects of these internal signal on the circadian rhythms. A similar study shows that when people are exposed to light at night time, a shift in their pacemakers [9] which is an evidence of the exogenous component of circadian rhythms. These exogenous components of the circadian rhythms have the ability affect positively and negatively our natural endogenous cycle. With light being what we are mostly daily exposed to, what is the influence of light on the circadian rhythms?

2.1.2 Internal circadian rhythms influenced by light

Melatonin is the hormone produced by the pineal gland in the suprachiasmatic nuclei (SCN) which has a soporific effect and the ability to entertain the sleep-wake cycle. While melatonin itself is not the cause of a person sleeping, it however creates changes in a person's body that affect their sleepiness.

The pineal gland responsible for the secretion of the melatonin hormone is under the influence of one of the biological clocks, the master clock located in the SCN. The SCN receives visual information from the retinal-hypothalamic tract which entrains the SCN according to the photoperiod [3]. The SCN in turn activate a gene in the pineal cell (CREM) which produces a protein (ICER) needed for the production of melatonin. As a result, the secretion of pineal hormone melatonin tracks the light/dark cycle with its secretion being high during the day and low during the night.

[1] it oxalates generally a period near 24h

2.1.3 Quantitative and qualitative characteristics of light on melatonin production

Many types of research have been done to understand the impact of light on the circadian rhythms especially the sleep-wake cycle.
Kathleen *et al.* in their paper *Blue light from light-emitting diodes elicits a dose-dependent suppression of melatonin in humans*[4], provide details information on their finding of the effect of light on humans subject. Subjects used for the experiments were 5 males and 3 females with a mean of 23.9 ± 0.5 years, with each subject demonstrating normal colour vision. The lighting requirement was blue light of $\lambda_{max} = 469nm \pm 1nm$ with $\frac{1}{2}$ peak bandwidth= $26nm$ and a typical viewing distance of $35cm$. The subjects were blindfolded from midnight to 2 AM. From thereon, they were exposed to 90 min of blue light exposure followed by another dark exposure. Blood samples from the subject were taken from 2 AM at 30 min interval.From their experiment, they concluded that:

- *Blue LED light has an increased melatonin suppression following an increase in exposure irradiance*
- *Blue LED light may have stronger suppressing effect than 4000K white fluorescent light.*

A similar study was previously made by George C. Brainard *et al.* [5] used 72 healthy human subjects, with normal colour vision. The subjects composed of 37 females, 35 males, aged between 18 and 30 years (mean age of 24.5 ± 0.3 years), came from different ethnic (African, African Americans, Caucasians, Asian, Hispanic). The melatonin suppression action spectrum was formed using eight different wavelengths between $440nm$ and $600nm$ *(440, 460, 480, 505, 530, 555, 575, 600)*. Using the same procedure as mentioned in the previous experiment, blood samples were taken at 30 min interval after 2 AM. The results of their research published in their paper *Action Spectrum for Melatonin Regulation in Humans: Evidence for a Novel Circadian Photoreceptor* the following conclusion:

- *Light irradiance greater or equal to* $3.1\mu W/cm^2$ *evoke a significant melatonin supression*
- *Smaller wavelength monochromatic light have a greater change in Plasma Melatonin as Percent Change Control-Adjusted for a fixed value of Photon density* ([5] pp. 4-5).

2.1.4 Impact of light on human behaviour and sleep-wake cycle

Among the Zeitgebers (natural phenomenon acting as a signal in the regulation of the circadian rhythm) of the circadian rhythms, light is the major Zeitgebers and has important effects on the human sleep-wake cycle. A study on the "Circadian and Light

Effects on Human Sleepiness-Alertness" made by Christian *et al.*[2] shows that with nor phase lag or lead between the circadian rhythm and the sleep-wake cycle, subjective sleepiness and core body temperature have opposite behaviour ([2], pp. 12, fig. 2.1). The human normal sleep-wake cycle is comprised of 8h of sleep and 16h of wakefulness [15]. This cycle is naturally affected by the human activities but is highly influenced by light exposure. The study shows that office workers with bright blue office light have a better sleep-wake cycle than those with dim and warm office light. Furthermore, it shows that light exposure of sufficient intensity at night can reduce the secretion of melatonin with alerting response starting within the first 20mm. With the recent advance in LED technologies, humans are no longer following the natural light/dark cycle causing numerous sleep disorders.

2.2 Lighting technologies

Light is capable of affecting the human sleep-wake cycle. However this impact can be used to reverse the negative impact of light on the sleep-wake cycle. This section introduces the different light technologies available and their use in regualting the human sleep-wake cycle.

2.2.1 Light

The sun's electromagnetic radiation has a broad light spectrum ranging from $100nm$ to $1mm$. After being filtered through the earth atmosphere, only certain portions of the spectrum are kept with the visible light spectrum having the maximum irradiance. Because the human circadian rhythms are automatically synchronised to the natural light-dark cycle, the characteristics of the sunlight are used as a benchmark in the emission of artificial light. These artificial lights are able to produce more or less the same wavelength as the sunlight but with much less irradiance.

2.2.2 Type of light technologies

Various light technologies have been developed over the years, their use and their usefulness in this project are detailed below.

Incandescent light

These are the most common and least efficient light technology. It produces light by passing a high current through a wire filament producing warm light as it glows. These type of lights produce only a specific spectrum of light (warm light) besides they are not designed to be programmable.

Fluorescent light

Light is produced by passing electricity through mercury vapour, the invisible light produced as a result of that interaction, connect with the coating of the glass emitting light. It produces all type of white light (warm, cool, daylight) with a good colour rendering. Compared to the incandescent light, it only produces one type of colour and it is not designed to be programmable.

Halogen light

It shares similarities with the incandescent light except that halogen gas is added to the glass. It produces a crisp white colour. As the previous technologies, it also produces one type of colour and it not designed to be programmed.

Xeon light

This type is another version of incandescent light with Xeon gas added to the glass instead, it produces a less yellow light. As the previous technologies, it also produces one type of colour and it not designed to be programmed.

LED light

This technology uses the passage of current through a diode to produce light. These light are very efficient and provide all sort of colour as well as cool and warm light. Moreover, their ability to work based on the passage on current allows these lights to be programmable. New LED technologies have a microcontroller which can be programmed the LEDs.

2.2.3 Light treatment of sleep disorder

With the discovery of the effect of light on the sleep-wake cycle, electronic devices producing specific light have been used in treating patients with sleep-disorder. Advanced sleep phase disorder (ASPD) have been treated using a therapeutic approach involving chronotherapy and timed light exposure [18]. The same concept has been used to create a home bed lamp or alarm clock facilitating the regulation of the sleep-wake cycle. **GE Sol** and **Philips***(which is actively involved in sleep-wake cycle treatment)* have created devices able to "influence" the human sleep-wake cycle.

C by GE Sol

C shown in fig. 2.2 is a "all-in-one smart light" [16] which has Amazon intelligent personal assistant Alexa built in. C has a various range of colours which are manually

selected based on the user preference. It is capable of communicating with GE sol devices and smartphones inserting it among the user's network of devices. Despite its high technology features and its elegant design, C remains just a bed lamp as it is not clinically proven to be helpful in regulating the user's sleep-wake cycle.

Figure 2.2: C by GE Sol, an intelligent lamp bed using Amazon Alextra.

Philips wake-up lamps

Philips has created a broad range of wake-up lamps designed to impact the sleep-wake cycle of the users. Figures 2.3a and 2.3b are examples of the recent versions of Philips' clocks able to simulate sunrise and sunset which last from 20 to 40 mm. These simulations vary the colour of the light following the sun's natural sunrise colour and end with a selected channel or prefered user's music. These Sleep and Wake-up lights are the only wake-up lamp clinically proven to work as stated by Philips[17].

NEED SOME TRANSITION HERE

2.3 Hardware modules

This section provides the research on the hardware technologies as well as the reasons behind the choice of some of these technologies.

(a) HF3531/60, coloured sunrise simulation, 7 natural sounds, Tap snooze and reading lamp, midnight light function

(b) HF3551/60, coloured sunrise simulation, 7 natural sounds, Tap snooze and reading lamp, midnight light function, Operated by iPhone App

Figure 2.3: Wake-up light by Philips

2.3.1 Processors and microcontrollers

A microcontroller is an integrated circuit which is dedicated to execute tasks of a specific application. It is physically small, cheap compared to a computer, and is designed to operate a low power consumption [19].
Numerous microcontrollers were analysed for this project. Each manufacturer provides various range of microcontroller suitable for different applications. The NPSC's microcontroller needs to communicate to a smartphone application wirelessly and control various external modules. While controlling the external modules, the microcontroller needs to perform processing of the sunrise and sunset patterns by continuously organising and sending data to the neopixels. In order to reduce the complexity of finding the *perfect* microcontroller for the NPSC, the selection was based on the following characteristics:

- **Size of the FLASH:** How many lines of code can be loaded to the microcontroller?
- **Cost**
- **Clock speed**
- **Community:** Does the manufacturer have a large community of developers?
- **Bit precision:** Are we aiming at 8, 16, or 32 bits precision?
- **Familiarity:** How familiar are we with the microcontroller (time is a constraint)?
- **Number of pins:** How many ports does it provide?
- **Extra features:** What are the built-in functionalities (wifi module, bluetooth)?

Table 2.1 provides the difference between the microcontrollers selected, this table was use to make the final decision on the microcontroller selection.

Table 2.1: Comparison between specifications of the Arduino Due [20], the Intel Edison [21], the Raspverry Pi Zero [22], and the STM32F407VGT6[23]

Characteristics	Microcontrollers			
	Arduino Due	**Intel Edison**	**Raspberry Pi Zero**	**STM32F407VGT6**
Clock speed	84MHz	dual-core, dual-threaded 500 MHz CPU	1GHz single core	168 MHz
Bit precision	32	32	32	32
FLASH	512KB	4GB	MicroSDHC	1MB
Pins	54	40	40	82
Cost	R549.25	R687.87	R79.8	R114.83

2.3.2 Storage

Storages are important in the development of embedded solutions. The **Electrical Erasable Programmable Read-Only Memory** (EEPROM) is a non-volatile memory capable of keeping its data after being powered off. There are two types of EEPROMs, serial and parallel EEPROMs. In a study made by Microship on the difference between serial and parallel EEPROM of 16KB, Tom Tyson from the Memory Product Divisions concluded that the serial EEPROM is *the best option for embedded solutions requiring a small EEPROM footprint, low current and low operating voltage, the ability and ease to programme a byte at a time, and the best price-performance non-volatile memory solution available*[24] (pp. 4). The NPSC needs to store the user's data and the NPSC default's settings. Having a non-volatile external memory easy to program is of benefits to the NPSC.

2.3.3 Wireless technology

Different types of wireless technologies allow devices to communicate to each other wirelessly. The Institute of Electrical and Electronics Engineers (IEEE) has grouped them in the 802.15 technologies. Among these are the well-known Wifi, cellular machine to machine (M2M), mesh network using ZigBee, Z-wave ... and Bluetooth.
The NPSC needs to make a wireless connection to communicate with a smartphone application. The appropriateness of the technologies tabulated in **??** for the NPSC are analysed below. Wifi modules are inexpensive; however, smartphones communicating with the NPSC will require being connected to the same wifi. Moreover, a web application might need to be designed as a platform for the NPSC, this will raise security issues that we would not be able to explore to the time constraint of the project. Cellular machine to machine has one main drawback, its recurring cost. As its name indicate, this technology is to send information from a machine to another machine and is not optimal for a graphic design platform[2]. Mesh networks are optimal for interactions of device/machine of the

[2]having a phone or web application that make use of this would be horrible

same kind (not a requirement), the NPSC does not fall into that category. Bluetooth is designed for short-range communication (the typical distance is around $10m$ [3])being in almost all smartphones it has been extensively used by embedded systems device such as handsets, Bluetooth speakers. It has a full-duplex communication with synchronous and asynchronous channel. Using Bluetooth, the NPSC will not require any network or incur any recurring cost to the user.

Table 2.2: Comparison between the constraints and the convenience of Wifi, M2M, Mesh network, and Bluetooth

Technology	Constraints	Convenience
Wifi	Wifi network and security protocol	Yes
M2M	Recurring cost	No
Mesh network	Machine/Device of the same kind	No
Bluetooth	Short range	Yes

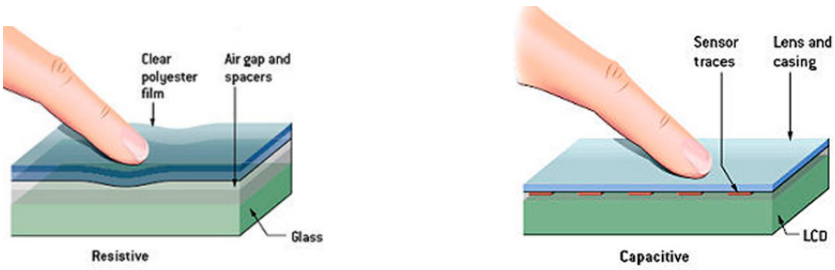
2.3.4 Touch screen

A touch-screen device can locate the position of a point of contact on its screen. Figure 2.4 illustrates the difference between resistive and capacitive touchscreen. Resistive touch-screens are made out of many layers of which two are composed of indium-tin-oxide (ITO) which is highly resistive and transparent. By applying pressure on one of the layers, the layers come in contact creating a signal that is used to find the location of the point of contact. Capacitive touch-screens, on the other hand, make use of the conductivity of the object in contact with the screen to affect the electrostatic field between the ITO layers. Resistive touch-screens are less complex than capacitive touch-screens, thus cheaper. Additionally, because they rely on pressure, any object whether conductive or not can be used on the screen which is made to be robust. However, resistive touch-screens do not support multi-touch and have poor contrast because of the extra layer used to protect the ITO layers.
The NPSC need to have an onboard controller. This controller must be user-friendly, with the different features of the NPSC, a touch-screen is desirable over a controller with physical buttons. A resistive touchscreen would add to the robustness of the NPSC as a whole.

2.3.5 Neopixels

The neopixel is a programmable light source using an MCU to control an RGB or RGBW LEDs. Each colour is capable of producing 255 brightness levels resulting in 16777216 different RGB colours. The light characteristics of the neopixel of choice are presented in table 2.3. The light requirement of the NPSC is the emmission of light of $460nm \pm 10$ (blue light) at an illuminance of $30lux$ minimum.

[3] can be increased by increasing the transmission power



(a) Resistive touch-screen technology[28]. (b) Resistive touch-screen technology [29].

Figure 2.4: Touch-screen technologies. More pressure need to be applied on the resistive touch screen for location dectection.

Table 2.3: Light specification of the WS2812 neopixels

Emitting colour	**Model**	**Wavelength (nm)**	**Luminous intensity (mcd)**	**Current (mA)**
Red	13CBAUP	620-625	390-420	20
Green	13CGAUP	522-525	660-720	20
Blue	10R1MUX	465-467	180-200	20

The illuminance of a light source at a point is given by:

$$E_{v(lx)} = \frac{I_{v(cd)}}{d_m^2} \tag{2.1}$$

eq. (2.1) is the illuminance directly in front of the light source. **Lambert's Cosine Law** says that the illuminance *is directly proportional to the cosine of the angle made by the normal to the illuminated surface with the direction of the incident flux.* [38], mathematically it means:

$$E_{v_1} = E_v * \cos(\theta) \tag{2.2}$$

with θ, the angle between the direction of the incident light and the surface normal.
The dimension to be considered for the calculation of the illuminance of the neopixel ring are shown in fig. 2.5. Using the dimension from fig. 2.5 and the lowest intensity emitted by the blue led, the total illuminace of the rings at a specific distance can be calculated using the following equation:

$$E_{v_{total}} = N * Ev * (\cos(\theta_1) + \cos(\theta_2) + \cos(\theta_3)) \tag{2.3}$$

The result of the calculation are tabulated in the table 2.4. The NPSC will not meet the light requirement if the it is placed more than one meter away from the subject.

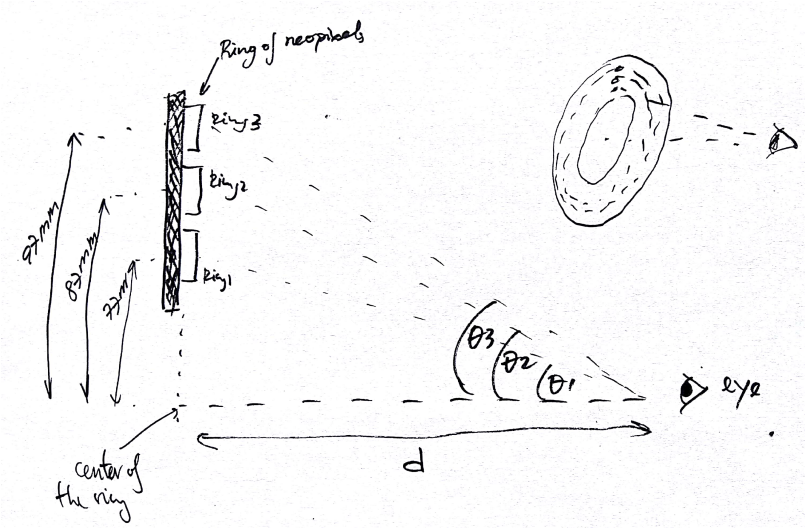


Figure 2.5: Relative position of the neopixel rings

Table 2.4: Illuminance of the Blue led of neopixel ring for distance ranging from 10cm to 110cm. Result with no angle consieration represent all neopixels in one point while results with angle consideration take into account the relative position of the each pixels to the subject location.

Distance (cm)	Illuminance (lux)	
	no angle consideration	with angle
10	3600	2715.18
20	900	824.76
30	400	384.02
40	225	219.8
50	144	141.84
60	100	98.85
70	73.47	72.9
80	56.25	55.92
90	44.44	44.24
100	36	35.86
110	29.75	29.66

2.4 Communication protocols

The NPSC requires different integrated circuits using various communication protocols, below is a brief on the use of the protocol required.

2.4.1 Serial Peripheral Interface (SPI) Bus

SPI is a synchronous full duplex serial communication protocol used for short distance communication of electronic devices. With the protocol, one master can communicate to many slaves using a chip select pin (use to select the slave) but a slave can only talk to the master device. SPI uses 4 signals namely, the Master Out Slave In (MOSI), the Master In Slave Out (MISO), the clock (SCK), and the slave select or chip select

(SS). Since SPI uses a clock, it does not require the configuration of a baud rate before communication. The main inconvenience with SPI is the number of pins required for a one to one communication, for every additional slave, the master must provide an SS pins which make SPI not the ideal protocol for communicating to multiple slaves devices [30].

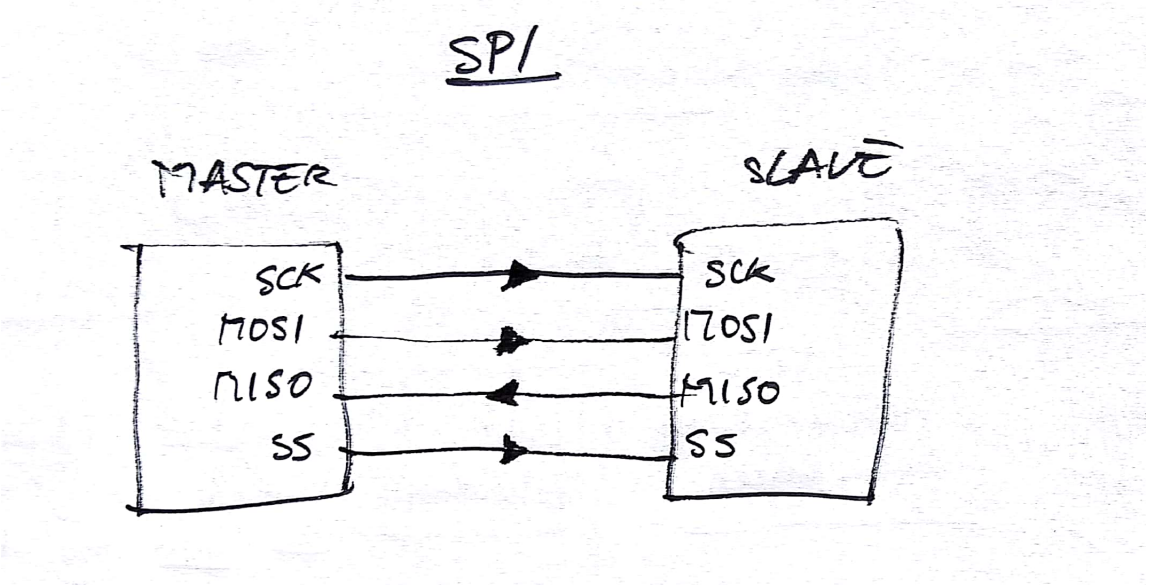


Figure 2.6: Wire connection setting for SPI communication between a master and a slave device

2.4.2 Universal asynchronous receiver-transmitter (UART)

UART is an asynchronous full duplex communication protocol making use of two signals a transmit signal (TX) and a receive signal (RX). With UART, both devices need to agree on a baud-rate for communication, this rate defines the number of bytes to be sent and received. The problem with UART is the complexity of the protocol required to ensure synchronous communication and correct transfer of information between device. Although theoretically, UART baud-rate is infinite, it is practically limited to 230400 bits per seconds [31].

2.4.3 Inter-Integrated Circuit (I2C)

I2C is a serial protocol that allows communications from multiple slaves to multiple masters. It takes a bit of both SPI and UART by being designed for short distance communication and requiring only two pins, namely the data line (SDA) and clock line (SCL). Its clock ranges from $100kHz$ to $400kHz$ and a byte is sent at a time. Furthermore, with the I2C protocol, each slave must have a unique address used by the master for communication [32].

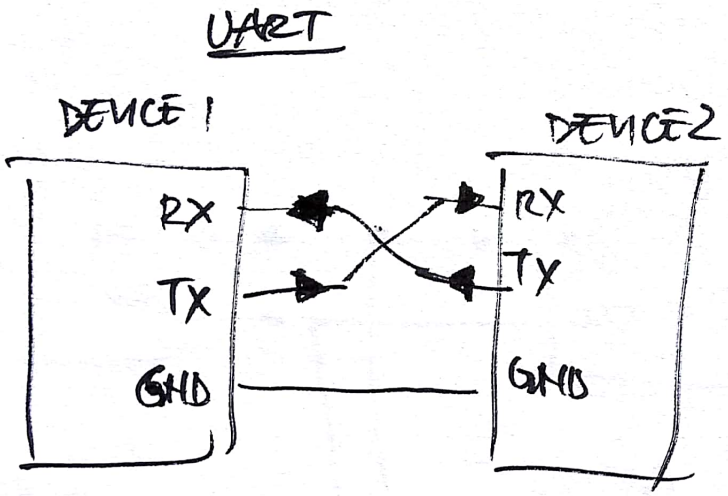


Figure 2.7: Wire connection setting for UART communication between two devices

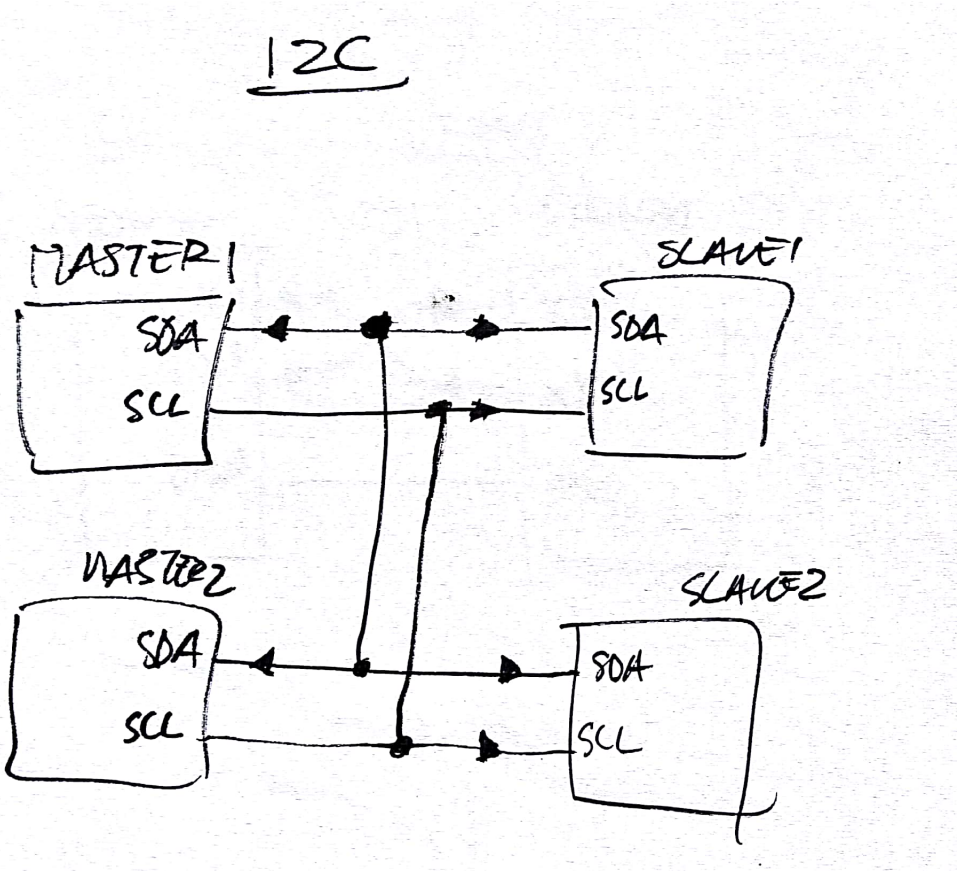
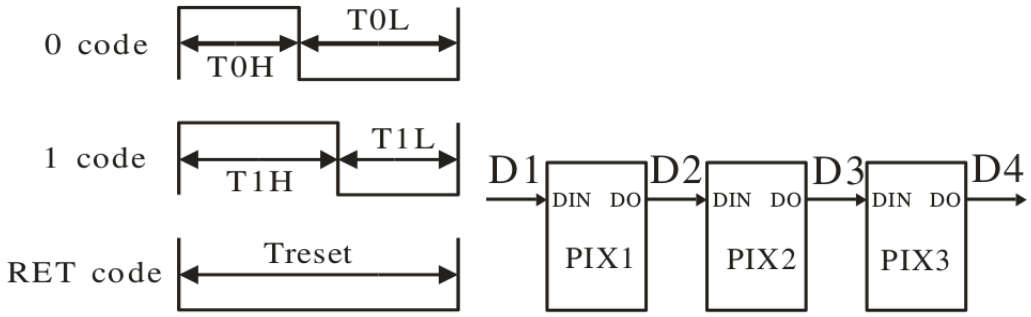


Figure 2.8: Wire connection setting for I2C communication between two masters and two slaves devices

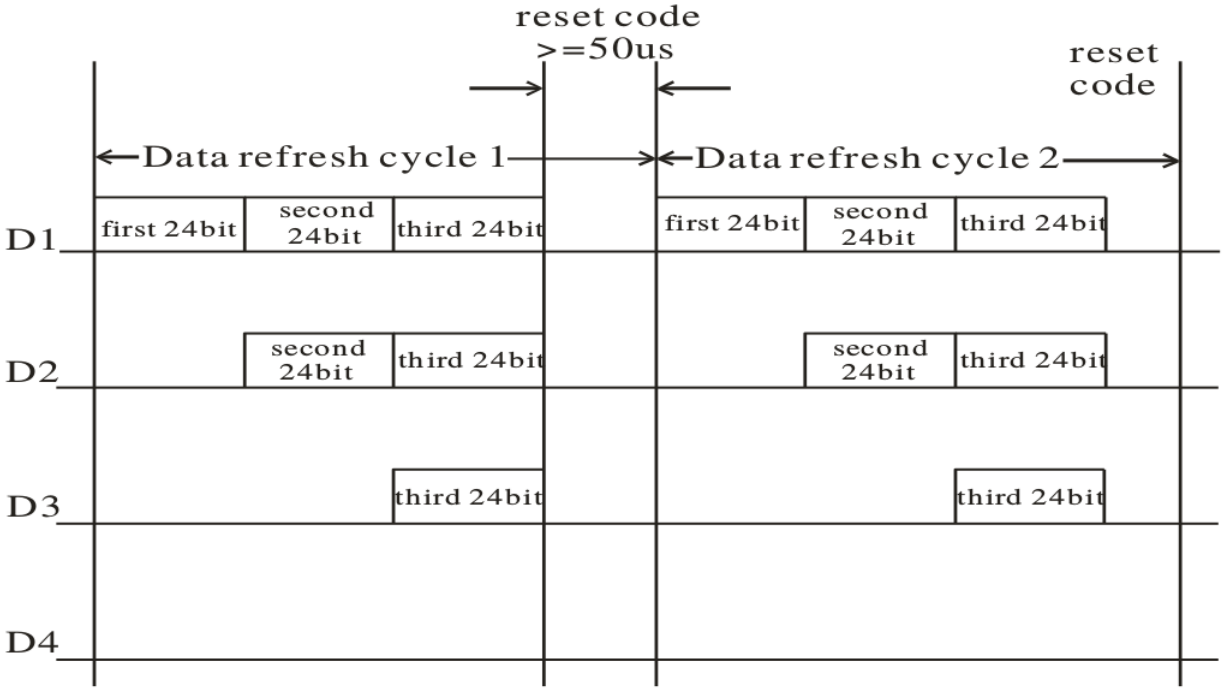
2.4.4 Neopixels serial protocol

Each neopixel has a built-in IC controlling the LEDs' intensity based on the data received. The neopixels use a single wire communication protocol. The neopixels can be connected to form a daisy chain (cascade)fig. 2.9c allowing to program of a series of neopixels using one signal. Each neopixel requires 24 Bytes (32 Bytes for RGBW LEDs), each bytes is an encoded bit using a Non-Return-to-Zero encoding fig. 2.9a. On receiving a sequence of data, the first neopixel in the chain takes the first 24 bytes and passes the rest of the data to the next neopixel in the daisy chain and so for. This operation continues until a *rest signal* is received by the first neopixel fig. 2.9c. This is possible because each neopixel is capable of reshaping the incoming signal preserving the integrity its integrity of continuous transmission.



(a) Non-return-to-zero encoding of a single bit in the neopixels programming protocol

(b) Neopixels connected in daisy chain (cascade).



(c) As the first neopixel receives a stream of multiple 8 Bytes chunck of data, it takes the first 8 Bytes from the stream and transmit the rest to the next neopixel in the cascade. A rest signal indicates where the stream of data ends.

Figure 2.9: Illustration of the neopixels serial interface

2.5 PCB Board Design

Printed Circuit Boards (PCBs) are almost present in every electronic circuit as they hold all the components together and implement the electrical connections. Designing a PCB is a process easily prone to errors, therefore a good PCB design requires the implementation of the design steps. Figure 2.10 illustrates an examples of the ideal pcb design steps. In making the NPSC PCBs these steps give an engineering approach that can be elaborated further based on the PCBs requirement. For example, the designer might question what is the best placement for certain component and thus be referred to some PCBs design standards such as the IPC2221.
One important design requirement (need) for the NPSC light requirement is its theoretical current drawn at full pixel brightness. The board-level diagram of the NPSC ring of the neopixel board should therefore be designed so that the track are wide enough to dissipate all the heat evenly across the board. The **IPC2221**, a generic standard for printed board design published by the *Association Connecting Electronics Industries* [33] describes a method for finding the mininmal width track given a specific current (see fig 6-4 from the IPC2221 [33](pp. 41)). A Javascript program based on the IPC2221 standard can be easily use to determine the track width [34]. This web application use the track cuurent, thickness, temperature rise, ambient temperature and track lenght as input to determine the required track width.

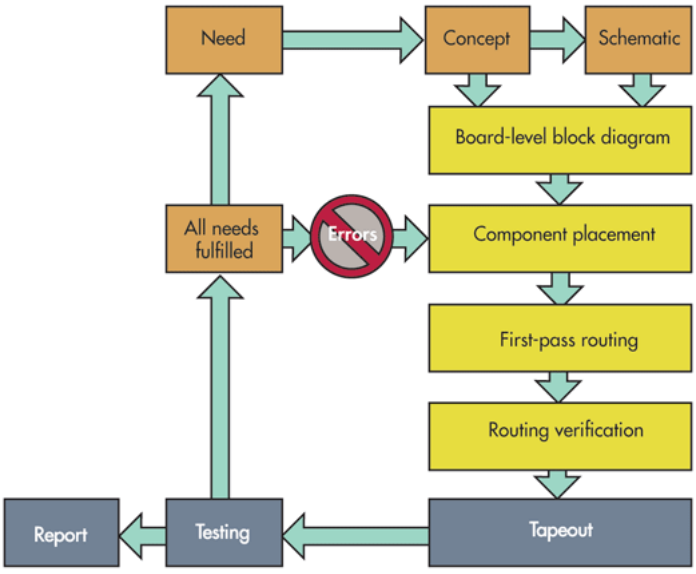


Figure 2.10: Ideal PCB design flow, starting from the need, followed by the design, implementation and testing

2.6 Programming Languages

Choosing the right programming language for software development is a crutial step, expecially in embedded system design as carefull manipulation of memory is required.

The languages below are the ones chosen for the development of NPSC.

2.6.1 C

C is a mid-level programming language [4] providing the ability to get close to the hardware while keeping some abstract layers for programming [35].It has an easy and straight forward syntax compared to languages such as Java, C has accumulated a lot of support and has lots of functions and documentation. Although C does not support Object Orientated Programming, it is a Procedure Orientated Language (POL) which means it is designed to create programs that follow an algorithm. This latter feature of C make it the favourite programming language for Embedded System.

2.6.2 C++

C++ is a programming language based on C which has higher level functionality. In the context of this project, C++ is used to test the functionality of the NPSC modules using an Arduino.

2.7 Software Tools and Libraries

Developing an embedded system software can be quite frustrating, for this reason tools with decent debugging features and programming interface should be chosen to reduce development to its minimal.

2.7.1 Atollic TrueSTUDIO for ARM

Atollic TrueSTUDIO is an IDE based on Eclipse. It comes with the GCC toolchain for ARM and with debugging features beuilt on top of GDB. It has more advanced features compared to Eclipse in the development embedded system application. Its debugging features include the use of any debug probe compatible with the GDB-server, P&E Micro, SEGGER J-Link / J-Trace, ST-Link, OpenOCD are all supported by Atollic TrueSTUDIO. It supports debugging of single and multi core devices and allows real time view of memory mapping, peripheral registers, advanced visualisation of variable and complex break point, CPU fault analysis and many more. Another important debugging feature of Atollic TrueSTUDIO is the instruction tracing and system analysis real time event analysis of Real Time Operating System [39].

[4] has the advantages of both ligh and low level language

2.7.2 Nextion IDE

Nextion IDE is the official IDE designed for programming any Nextion HMI touch-screen. The IDE allows the designed and programmation of a GUI interface and the definition of serial commands to be sent to a MCU. Nextion IDE is designed to reduce significantly development time of touch-screen interface in an embedded system project [40].

2.7.3 MIT App Inventor 2

2.8 Design Models

Designing an embedded system requires the use of suitable design methods. This section describes the different design models used for the project.

2.8.1 V-Model

The V-Model is a linear product-development methodology composed of two main parts. On the left side of the diagram (see fig. 2.11) is the project definition. On that branch, the requirements are defined, the system design, architectural design and module design are performed. At the bottom of the diagram is the implementation of the design. Following the implementation is the project testing and integration. The project testing starts with the unit testing of the module design followed by integration testing in which the architectural design is tested to ensure that the system functions well across all components. The next step consists of the system testing and the validation of the performance defined in the design. The last step is the acceptance testing, done to ensure that the system can be deployed.

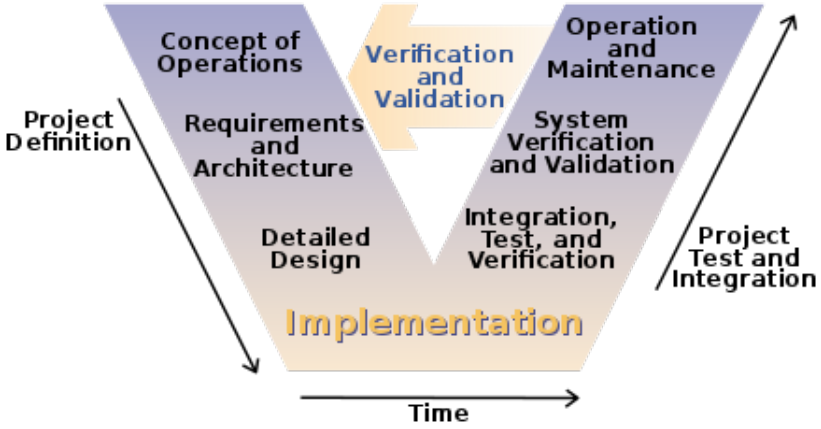


Figure 2.11: V-Model basic template used for system definition and testing

2.8.2 Spiral Model

The Spiral model is a combination of the iterative model [5] and the waterfall model [6] through which a product can be refined after each cycle. Figure 2.12 illustrate the order of the spiral model steps. The first step is the identification, it consists of identifying the user, system, sub-system, and unit requirements. The second step consists of the system, architectural, physical design and logical design of all modules. The prototype is built during the third step, the prototype serves as a proof of concept. During the last stage, the prototype performance is evaluated as well as the validation of the system requirements. Moreover, a risk and cost analysis alongside with the overall feasibility of the project is evaluated at this stage.

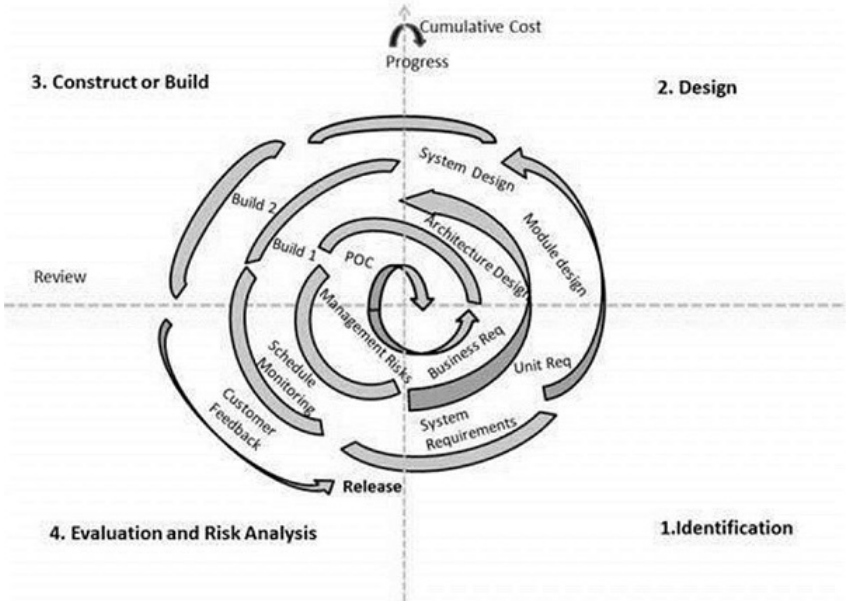


Figure 2.12: Spiral-Model basic template used for system refinement

[5] Cyclic process consisting of design, prototyping, testing, analysis and refinement of a product

[6] linear sequential process consisting of conception, initiation, analysis, design, construction, testing, deployment and maintenance of a product.

Chapter 3

Methodology

This chapter describes the way the research for the project was done. It is explained with sufficient details such that the device can be rebuilt again by anyone reading the report and having some basics understanding of electronics and embedded system programming.

3.1 Outline

The project is about the design of an embedded system capable meeting the light requirement as mentioned in chapter 1 while being user-friendly and personalisable to the user's content. To achieve this, two design methods were used. The spiral model was used to provide an understanding of the cycle of the ES, from the identification of the requirement to the validation of the user requirements. As for the V-model, it was used to ensure a thorough testing of the device ensuring that all modules and system at a higher level function perfectly.
MENTION SPIRAL MODEL USE HERE.
The first step in the development of the design consists of the understanding of the requirements presented in section 1.3.

3.2 Literature Review

The review of research related to the internal clocks, how are their endogenous influencers and what technologies can be used to affect these clocks are made to get an understanding of the problem. This is not only limited to gathering information about the theory involved to meet the requirements but it also covers the reviews of documents and articles on hardware and software modules and tools that could possibly be of use.
Below are the questions by which the literature review was driven.

1. What are the problems? This step consists of seeking information about sleep disorder and its causes.

2. What biological mechanisms lead to the problem? Here research papers on biological clocks and circadian clock and rhythms were analysed to see their relationship with the problem.

3. What are the characteristics of the light causing sleep disorder? When does the reception of these lights cause a problem? A this point, there was a correlation between light and sleep disorder in humans. In order to control the effect of light on patients having sleep disorders, the characteristics (wavelength, incandescence, ...) of the light are required.

4. What technologies can be used to mimic these light characteristics? This step involved a careful analysis of all artificial light source technologies available.

5. What device has been created to tackle the same problem? An analysis of the competitors' devices.

6. What hardware is required by the device?

7. What communication protocol is used by each hardware?

8. What software tools are available for the development of the device software?

9. What design methodology is suitable for this type of project?

3.3 Setup

This section describes the tools to be set prior any implementation and design. The setups are divided into two sections; the hardware setup and the software setup.

3.3.1 Hardware setup

Below are all the hardware modules required for the development of the NPSC.

Personal or Desktop Computer

A computer is needed to design, program and debug the hardware. Most of the softwares run on Windows, Linux and IOS but it is recommended that a Windows computer running on windows vista minimum is used for the development of the project.

STM32F4 Discovery board

The STM32F4 Discovery board is the evaluation board of the STM32F407VG. It has an onboard STLink debugger to facilitate the programming of the STM32F407VG and provides all the I/O pins to the microcontroller.

Arduino board

The library made for the STM32F4 are all custom made, there is a risk of being stuck with debugging these libraries as it might be thought that the modules are not working and not the libraries. Arduino boards have libraries proven to work for most of the modules used in this project. The role of this board is purely for testing the functionalities of the modules.
Note that the use of this board is optional. Any Arduino board can be used; however, an Arduino Uno was used in this project.

Electronic modules

All modules mentioned in section 5.3 are required. They are all interconnected and the absence of one can cause the whole system to fail.

Logic analyser

A logic analyser is a tool that analyses the data coming from one of its channel based on predefined communication protocols. This is a very useful device in debugging the modules interaction with the microcontroller. The logic analyser used for this project is one of the Salea logic analysers.

Figure 3.1: Salea logic analyser.

Serial to USB converter

The Serial to USB converter allows the translation of UART commands to USB command. This module is required to program the Nextion screen.

3.3.2 Software setup

Certain software tools are required for the development of the NPSC software. Those mentioned below are the one used for this project, other version or type of software can be used as long as they have the same functionalities as the one listed below.

Atolic TrueSTUDIO

Atolic TrueSTUDIO provides debugging functionalities such as breakpoint, memory and register observation as well as backtracking. It contains all the STM libraries which facilitate development. Moreover, it comes with STLink so no external debugging setup is required. Other IDEs such as Keil or Eclipse can also be used instead of Atolic TrueSTUDIO. In such case, it is essential to make sure that C is installed as well as gdb and STlink alongside its drivers.

Nextion IDE

The Nextion IDE is the only IDE capable of programming the Nextion TFT touchscreen and only runs on windows. It provides a Graphical User Interface when visual components can be dragged to the screen. Each type of component has specific parameters and events. The IDE also allows the user to program the behaviour of a component depending on its events. Before uploading the program to the Nextion screen, the user should use the debugging mode to test the logical behaviour implemented and the data transfer of the screen.

Logic analyser software

The logic analyser mentioned in section 3.3.1 requires the use of the Salea software which runs on Windows, Linux and IOS. The software allows the selection of up to eight channels whose data can be visualised and analysed. Before analysing the data, the sampling rate and time period must be set based on the baud rate or frequency of the pin to debug. After the first setup, an analyser is set up to the desired communication protocol. In this project only the protocols mentioned in section 2.4 are needed. The data can be collected and automatically analysed once the setup stage is done.

PCB design software: Altium Designer

A PCB design software is required in the design of the hardware. The one used for this project is Altium designer as the university has a working license and Altium has more sophisticated features than its competitors.

API generator: Doxygen

The interactions between the elements in the system become more complex for every hardware module, sub-system and system design added to the project. In order to keep track of these interactions, an API generator is required. An API generator generates the documentation of the system based on the comments in each file. Doxygen which is popular among the API generator, especially for ES is used for this project.

Version Control: git

The whole project is under a version controller. Each implementation type (hardware, software, report, mechanical, ...) has its own branch to control each change made in the project. This project repository is under git and can be found on github.com.

3.4 Implementation

The NPSC requires hardware and software modules. For the first prototype, certain hardware modules off the shelves can be used, as for other modules, they require to be designed and manufactured as modules with their specific characteristics do not exists on the market.
The implementation was broken into two main sections, the hardware implementation and the software implementation. The software implementation relies on the functioning of the hardware. To ensure that the system is implemented in a way that uses the time efficiently, it was implemented following the hierarchy illustrated in fig. 3.2. Figure 3.2 has five levels. Elements from the same level are able to call functions of the level below and not vice-versa. This implies that in order to implement one level, the level bellow must exist and function. The levels and their interactions are details as follow:

3.4.1 Level 1: Hardware and Utilities

The hardware and the utilities software are at the bottom of the hierarchy and are the first to be implemented. The hardware is manufactured and assembled, as for the off the shelves modules, a basic test using the Arduino is performed on the modules. The utilities software contains all the common functionalities or structures required by the element at higher levels. This element might not be implemented initially, in reality, this element is continuously modified based on the need of the higher elements.

3.4.2 Level 2: Framework

The framework element contains all the framework required by each hardware module. There is a one-to-one relationship between the framework modules and the hardware modules. The framework provides the basic functions required by higher modules. For

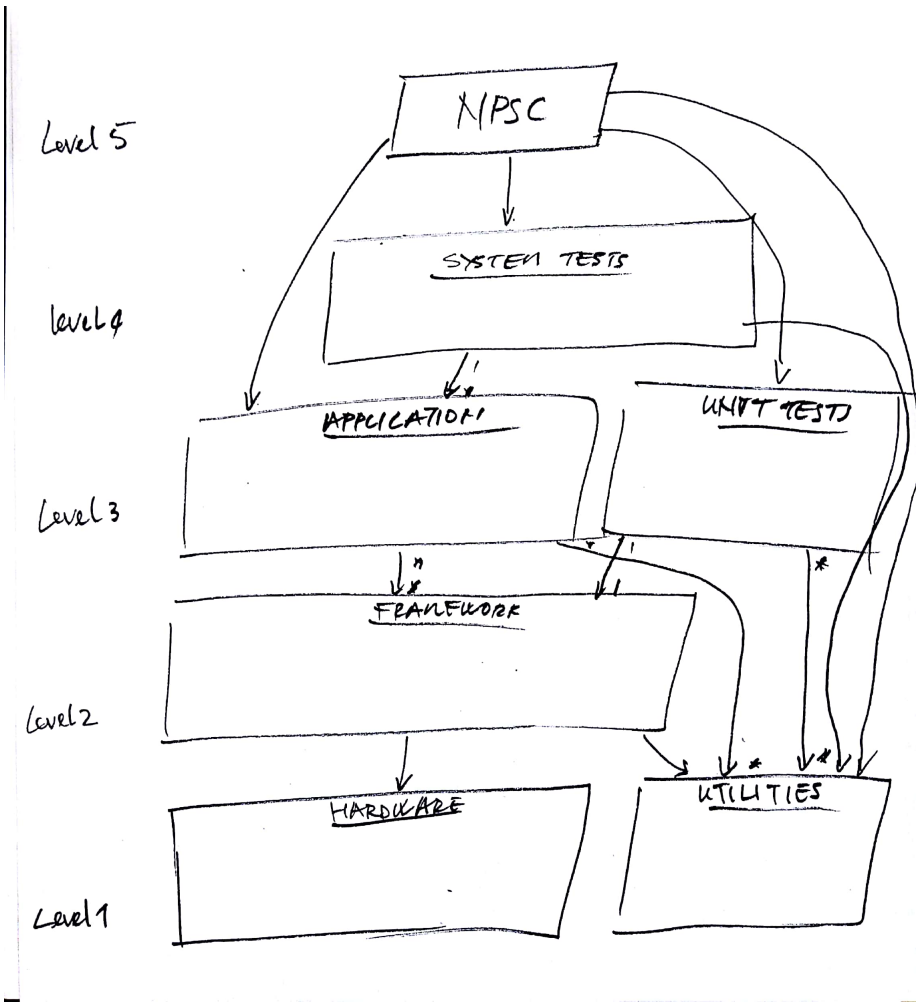


Figure 3.2: Overview of the implementation hierarchy. Implementation starts from the bottom with the collection or design and manufacturing of the hardware followed by the software implementation

example, each framework module provides an initialization function used to initialised the communication between its corresponding hardware module and the NPSC.

3.4.3 Level 3: Application and Unit tests

The unit test provides tests for each framework module. There is also a one-to-one relationship between the unit test modules and the software modules. The application element of the hierarchy contains all application codes required by the system. These applications have a one-to-many relationship between them and the framework modules. For example, the alarm application requires the internal RTC framework and the external RTC framework.

3.4.4 Level 4: System tests

The system tests contain the test designed to verify the system requirements. An example of a test is checking that a specific command from the screen executes the right command.

3.4.5 Level 5: NPSC

The NPSC is the highest level of the hierarchy. It, however, existed since the first implementation of the framework. It is used to call all the function needed for the application element and to run the unit and system tests.

3.5 Experimentation

This section presents the different tests designed to ensure that each hardware modules, sub-systems and system requirements are met. Each experiment targets one specific element from the hierarchy in fig. 3.2.

3.5.1 Hardware tests

The hardware testing requires careful test are any wrong connection could possibly damage the hardware. Two main tests are made on the hardware, a basic test testing the functionality of the hardware modules and the neopixel ring PCB test to ensure that the light requirement is met.

Basic hardware test

Most hardware communicates to the microcontroller through the protocols mentioned in 2.4. The test of these hardware is done using these steps:

1. Check connection to the micro using the hardware datasheets
2. Check power connections
3. Check functionality of module using benchmark testing (Arduino) if available otherwise use framework module to test hardware [1]
4. Use Salea logic analyser to verify the data received from the hardware is correct.

[1] Is it risky to use the framework module to test the hardware as it has not been tested at this stage

Neopixel ring test

The Neopixel ring has light and current requirements. The light requirement is tested using a lux meter available in smartphones. As for the current requirement, the current withdrawn by the PCB is measure are different light intensity.

3.5.2 Software tests

The software tests rely on the hardware, therefore, the hardware tests must be run prior these tests. There are two types of test performed, namely the communication protocol tests (coms tests) and the logic tests.
The coms tests are performed using the debugging tools of the selected IDE (Atolic TrueSTUDIO in this project). Breakpoints are used alongside the Salea logic analyser to verify the information transferred between the STM32F4 and the hardware modules.
The logic tests are performed during runtime or using the IDE debugger [2]. The logic tests make use of a visual signal (either via the Nextion screen or Android application or Neopixel ring) to output the outcome of the tests.
There are two levels of software tests, the unit tests and the system tests.

Unit tests

Unit tests are designed to test the framework of the hardware modules. These tests are logic tests performed on the hardware. One example of these test is verifying the data stored on the storage after a write operation, or getting the colour from one neopixel. Unit tests also include indirect hardware interaction functions such as functions which convert a structure of data into another.

Integrations tests

Systems tests are at a higher level than the unit tests. They are all logic tests ensuring that the interaction between the framework modules is as expected. When these tests fail, the unit tests of the involved framework modules must be tested. If the unit tests pass, it is certainly a logic error and the IDE debugger must be used to pinpoint the errors.

3.5.3 Performance tests

This tests gather data on the performance of each module and of the system as a whole. It is meant to answer the following question:

1. What is the optimal queue size for the instruction buffer?

[2]The IDE debugger is done to pinpoint the point of failure in the code. It might require the use of a com test and is used if the runtime test fails

2. What is the optimal DMA size for each input?

3.5.4 Acceptance tests

This is the final test done to prove that the device meets all the requirements defined in section 1.3. This test involves using the device as a user and verify the functionalities of the device.

3.6 Analysis

This section provides analysis of the tests performed during the experiments. **TODO**

Chapter 4

Preliminary Design

This chapter aims to provide a general overview of the design, the interaction between the different modules of the NeoPixel Sunrise Clock (NPSC). Moreover, it provides the justifications of components selection and the reasoning behind the visual design of the NPSC.

4.1 System overview

The NPSC is designed such that it is a user-friendly bedside alarm-clock serving as a supplement in the treatment of sleep-disorder via light emission. To achieve these goals, the system was designed using a modular approach. The modules illustrated in fig. 4.1 are the elements deemed required to achieve the goals laid out in 1.2.2.
The relationship between the goals defined in the introduction and these modules are given below:

- **Regulate the sleep-wake cycle**: This is done by emissions of specific lights parameters using the Neopixel as part of the system outputs.
- **User-friendly**: Using push-buttons for controlling the NPSC would not be optimal it is a multifunctional device. The onboard touchscreen in the System Inputs is to remove having the users relying on push-buttons to use certain functionalities. As for the android application, it allows the users to control the devices wirelessly. Via the Android application, the NPSC has the possibility to be an Internet Of Thing (IOT) device increasing the variety of its applications.
- **Personalisable**: The system inputs are the doors to the NPSC configurations, with both inputs being touchscreen devices, multiples functionalities can be implemented. The EEPROM is to store user-specific data such as alarm, visual preferences.
- **Alarm-clock**: The Clock module consisting of a Real Time Clock (RTC) and an Alarm are meant to interact together to make the NPSC also an alarm clock.

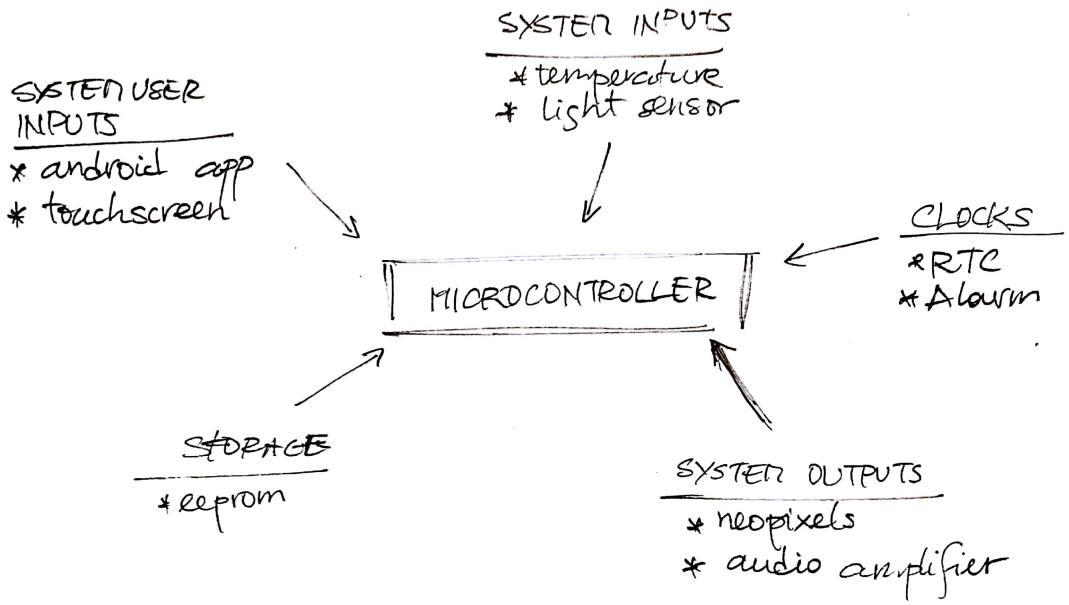


Figure 4.1: Structural block diagram of the NPSC.

4.2 System components selection

This section provides reasons for the selection of the components described in 4.1.

- **Microcontroller**: The STM32F407VGT6 with the specifications listed in table 2.1 is the one selected for the NPSC. This choice was based on its number of pins and mostly on the fact that this microcontroller has been used in previous projects.
- **Wireless**: As mentioned in 2.3.3, the preferable wireless technology is Bluetooth for its low complexity and the reliability of its connection. The HC-06 Bluetooth module has been chosen for this prototype for its popularity among microcontrollers application.
- **Eeprom**: There are no specific requirements on the EEPROM selection, for this prototype, only few information might require to be stored. It was assumed that a 32KB EEPROM would be enough. The EEPROM chosen is the 25LC640 64KB as it was proven to be reliable in previous projects. Moreover, the framework for this EEPROM was already developed for an STM microcontroller in the third year Embedded System project.
- **RTC**: The NPSC is an alarm clock and must be able to get accurate time. The STM32F4 series have a built-in RTC with two alarms. Because this prototype might not run on batteries and the RTC data would be lost if the microcontroller loses power, an external RTC module was required. An off the shelves breakout board for an RTC (the DS1307) is heavily used as an RTC for microcontroller applications and is used for this project as the external RTC. This breakout board powers the DS1307 with a battery that can last 10 years.

- **Light**: As mentioned in 2.3.5, the Neopixels were used as light source.
- **Touchscreen**: Many touchscreens on the market require the use of dozens of pins for their control. Besides, there are no common libraries for the modules increasing developing time. The touchscreen chosen for this prototype is the Nextion TFT touchscreen from Itead. It solves the issues mentioned above by providing an IDE (the Nextion IDE) used to program the screen via USB (required device from 2.7.2). It moreover provides an easy way of communicating with the microcontroller using two pins (TX and RX in the UART protocol). The model chosen is the NX4024T032_001, a 3.2" touchscreen with a resolution of 400x240 px.

4.3 Visual outputs component design decisions

The NPSC's visual impact is very important as it has great influence on the users. While focusing on the visual impact, the NPSC light source must meet the light requirements. To achieve both aesthetic and functionalities, the preliminary aspects were considered:

- Light emission of $30lux$: This is an important function of the NPSC. To achieve this, the NPSC must have a main light source module, based on table 2.4, 180 neopixels are sufficient to meet this requirement on a subject placed at $1m$ of the light source. For its aesthetic, the main light source is composed of three rings of 60 neopixels each. A great visual functionality could be displaying the hours, minutes and seconds one each ring, having the pixels on the seconds ring being light up as the second passes and so forth. Note that the board size needs to be large enough to dissipate the power consumed by the pixels (see 2.5).
- Alarm clock: What is the purpose of the clock that does not display the time? For this reason - and to fill the gap in the centre of the ring - three main pieces of information are to be displayed:
  1. Date and Temperature: seven segments RGB displays are used to display this information. Using RGB displays provides consistency in the choice of colour that the NPSC can emit.
  2. Time: A bigger version of the seven segments displays used in the point above could not be found, so the neopixels were used to simulate the seven segments RGB displays.
  3. Weekday: A row of seven neopixel, one for each day in the week.

4.4 Mechanical case

The NPSC must be able to stand on a bedside desk, these desks have a general size of $50cm * 50cm$. The NPSC ideal mechanical design is displayed in fig. 4.2.

Each side is meant to have specific use described below:

- **Front**: All visual outputs of the NPSC are on the front side. The main light source (the ring PCB) uses most of the front space. The other visual outputs PCBs are designed to fit inside the ring. The front side should be covered by a material hiding the PCB but letting most of the light pass. Here the NPSC should provide the look of a dark glass whose part can be illuminated.
- **Top**: The onboard touchscreen is located on the top of the NPSC. The touchscreen should be detachable from the NPSC case to allow the user to view the changes on the board while modifications are made.
- **Right and Left side**: Each side should have a speaker for the alarm. On the right side on the NPSC, a set of critical buttons are used to power off the device or stop the alarm.

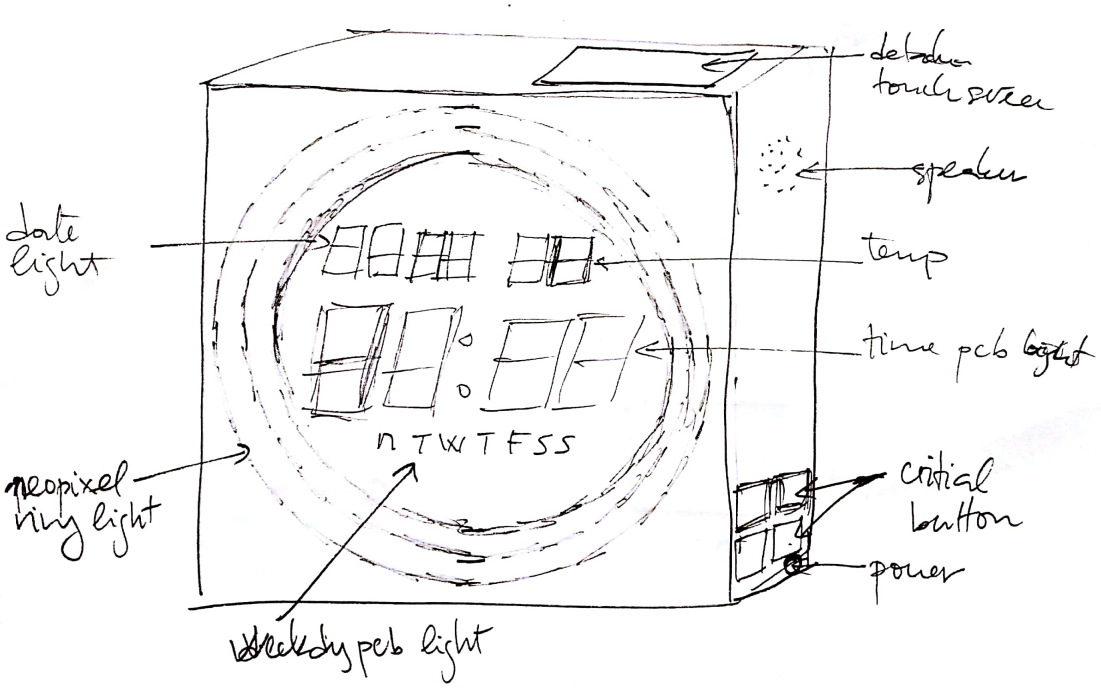


Figure 4.2: Mechanical prototype of the NPSC.

Chapter 5

Prototype Design

This chapter focuses on the design of the prototype of the NeoPixel Sunrise Clock (NPSC). It starts by providing an overview of the NPSC system design, followed by a high-level description of the sub-systems design and a detailed design of the hardware modules. The system design provides an overview of the NPSC's module. The high-level design focuses on the thinking process behind the design of the applications at level 3 of the system hierarchy. The detailed design focuses on the hardware modules at level 1 of the system hierarchy, providing more detailed on the design of the hardware.

5.1 System design

The system requirements of the NPSC are the following:

- The NPSC should be able to emit light and simulate sunrise.
- The NPSC should be able to display time and set alarms.
- The NPSC should be controllable using a touchscreen device.

These system requirements are achieved by the functions of the applications at level 3 of system modular hierarchy illustrated in fig. 3.2. Each application focuses on one requirement but might involve other requirements, the relations between some hardware modules and their application module are illustrated by fig. 5.1 and explained below.

- **Visual application**: This application controls the *Ring*, *Time*, *Weekday*, and *Date* PCBs. It manages visual outputs hardware, from setting the brightness to ensuring that a specific pattern is displayed to ensuring that the desired lux quantity is emitted by the NPSC.
- **Alarm application**: This application manages the synchronisation of the internal and external clocks. It is also responsible for managing the alarm stored in the EEPROM and updating the alarm.

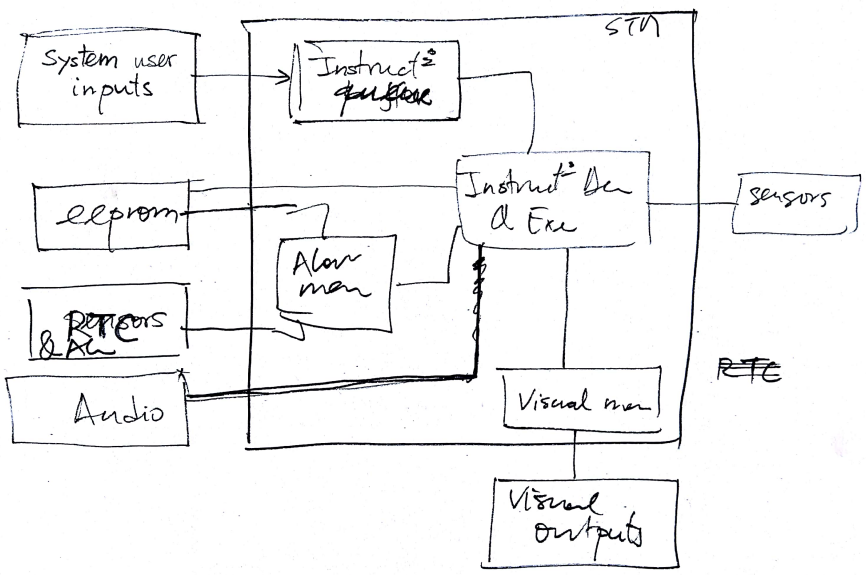


Figure 5.1: Detailed structural block diagram of the NPSC showing the connections between the hardware modules and the main software modules.

- **Instruction Queue application**: This application is responsible for the creation of instructions from the user input devices (Nextion touchscreen and Smartphone application) and manages the instruction queue.
- **Instruction Fetch Decode and Execute (IFDE) application**: This application relies on the result of the instruction queue application. The IFDE fetches instructions from the instruction queue, decode the instructions and execute commands based on the instruction's opcode. It is the bridge between the user inputs and the hardware modules of the NPSC.

5.2 High-level design

This section provides more detailed explanation of the system design requirements broken down into sub-system design requirements. The sub-systems requirements are listed below:

- Control light pattern
- Control light parameters
- Update and obtain time and date from clock
- Set, edit alarms
- Store user preferences
- Control the NPSC using an onboard touchscreen
- Control the NPSC using a smartphone application

The following sections explain the design of the main modules of the NPSC created to meet the system and sub-system requirements.

5.2.1 External storage

The storage's purpose is to store useful information such as the alarm parameters and the user's preferences. Certain applications require storage capability, therefore there is a restricted access to the EEPROM memory. The memory allocation is detailed in fig. 5.2.

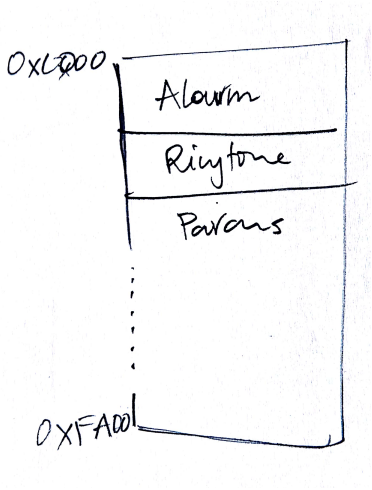


Figure 5.2: EEPROM memory allocation. The first section of the memory is allocated to the alarms, the second to the ringtones and the third section is allocated to the user parameters.

5.2.2 Inputs/Outputs instruction design

This section focuses on the design of the user inputs capture and interpretations. Figure 5.3 shows the communication between the microcontroller and the inputs, and the different step through which information sent by these inputs are analysed.

From the inputs to the microcontroller

There are two user inputs, a smartphone application running on an android device and a nextion screen. The android application communicates with the microcontroller via Bluetooth. Both the Bluetooth device and the Nextion touchscreen use the Universal Asynchronous Receiver-Transmitter (UART) protocol to communicate with the micro. Because the instructions are more than one byte, the microcontroller must be able to receive a stream of bytes with no information loss.
To achieve this, each UART uses the Direct Memory Access (DMA) to store each stream received to the corresponding DMA Buffer. After the reception of a stream of data, each DMA Buffer converts its data into instructions which are added to the instruction queue.

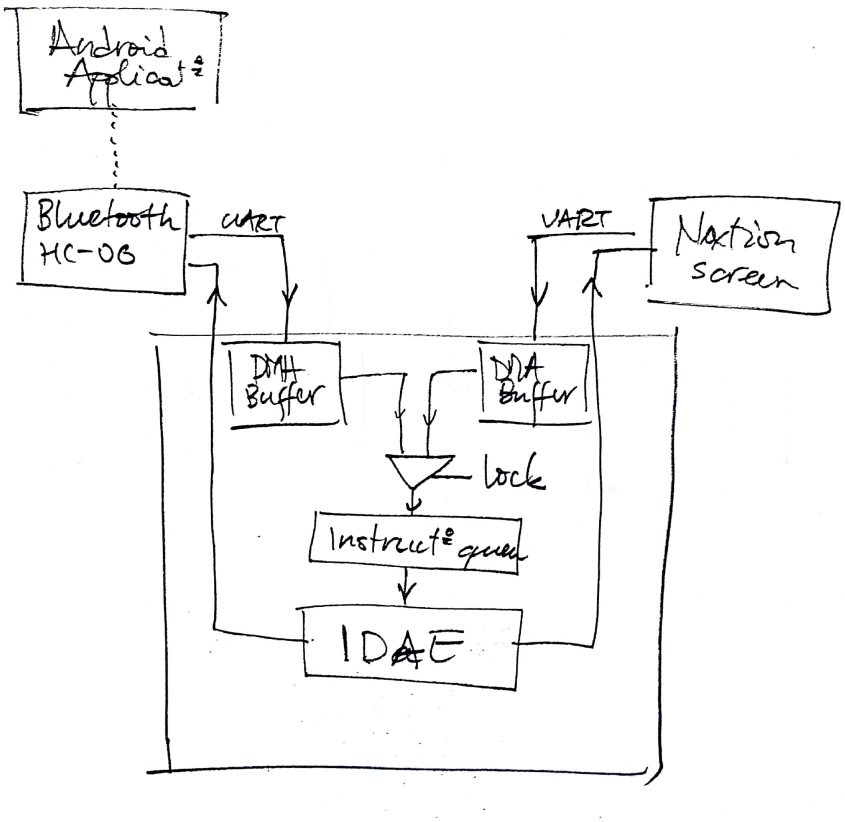


Figure 5.3: Strutural diagram of the hardware and software modules of the input and output instructions.

The DMA function is called on interrupt when the microcontroller receives data on the DMA channel. For this reasons it might occur that one DMA pauses the addition of instruction to the instruction queue by another DMA, and start adding new instructions to the queue. This is a potential concurrency problem that can corrupt the instructions added to the queue. To solve this problem, the addition of instruction to the queue is considered as a critical section of the design. A lock is thus added to the instruction queue such that only a process owning the lock can add new instructions to the queue. The instruction queue is a First In First Out (FIFO) queue, meaning that an element removed from the queue was the oldest element in the queue. This allows the instruction to be executed in order of arrival.

Instruction Fetch Decode Execute (IFDE)

The IFDE act as the Central Processing Unit (CPU) of the NPSC. Its role is to fetch instructions from the instruction queue, decode the instructions and execute them by calling the corresponding methods. It is the brain of the NPSC operation and acts as a bridge between the user inputs and the different hardware module. As the IFDE uses specific instruction set, there is a restriction on the actions performed by the users reducing any security breaches. By using an instruction set and the IFDE, the

functionalities of the NPSC can easily be expanded and modify during the software development.

Buffer size and Instruction sets

The instruction size was mostly defined by the rules of the Nextion touchscreen. This touchscreen is able to send information to a microcontroller using UART. However, it does not send a single byte of data unless a character is sent. As it is not convenient to convert all the integer parameters into characters, the second build-in method used by the nextion to send information was considered. This method sends an integer value as four bytes of data. Because certain methods of the framework require structs [1] as parameters, the instruction size was set to be double the size of the number of bytes sent by the nextion touchscreen per each method call.
The DMA Buffer size is dependent on the design of the Nextion touchscreen application and the Android application, as a stream of instruction might be sent. The number of instructions in the queue is also dependent on the DMA Buffer size because there are two inputs and therefore two DMAs, a rule of thumb for the instruction queue size is to be at least the number of inputs multiply by the DMA Buffer size.

The instructions of the NPSC are listed in table 5.1. The instructions are divided into categories, each category represent the actions performed at a framework level or application level (see fig. 3.2). There are two types of instructions, the set instructions and the get instructions. The set instructions write informations to the NPSC while the get information are just signal indicating that the user input device is requesting information. The instrcutions are 8 bytes long and the first byte represent the opcode of the instrucion while other byte are for the data sent. Instructions of the same category have the same most significant four bytes, for example all intructions from the Alarm category have $0x1*$ as their opcode with $*$ being any one digit hexadecimal number. The pcode 0x00 is reserved for identifying data that must not be converted into instructions.

[1] A struct also known as a record is a collection of data type used to represent entities having multiple attributes

Table 5.1: Instruction set used for the control of the NPSC by the users. Each instruction has a category, a name and an opcode. **-** in the table indicates data that will not be used by the IFDE. ***** indicates character data.

Instructions									
Category	Name	Instruction bytes							
		M0/opcode	**M1**	**M2**	**M3**	**M4**	**M5**	**M6**	**M7**
External RTC	set clock	**0x01**	year	month	date	weekday	hour	minute	second
	get clock	**0x02**	-	-	-	-	-	-	-
Alarm	set alarm time parameters	**0x10**	id	hour	minute	date/day selection	day/date	repeat	-
	set alarm extra parameters	**0x11**	id	ringtone	pattern	enable	fetch	-	-
	set alarm label	**0x12**	id	label number	-	*	*	*	*
	set alarm enable	**0x13**	id	enable	-	-	-	-	-
	get alarm minimum parameters	**0x14**	id	id	-	-	-	-	-
	get alarm parameters	**0x15**	id	-	-	-	-	-	-

5.2.3 Input devices

HOW ARE THE USER APPLICATIONS DESIGNED?

5.2.4 Alarm and Clock

The structural diagram of this module is shown in fig. 5.4. The microcontroller used by the NPSC has a built-in RTC with two alarms (alarm A and alarm B), this module makes use of an Alarm manager (application software) which control the EEPROM, the external and internal RTCs and the two alarms. Alarm A and B are triggered based

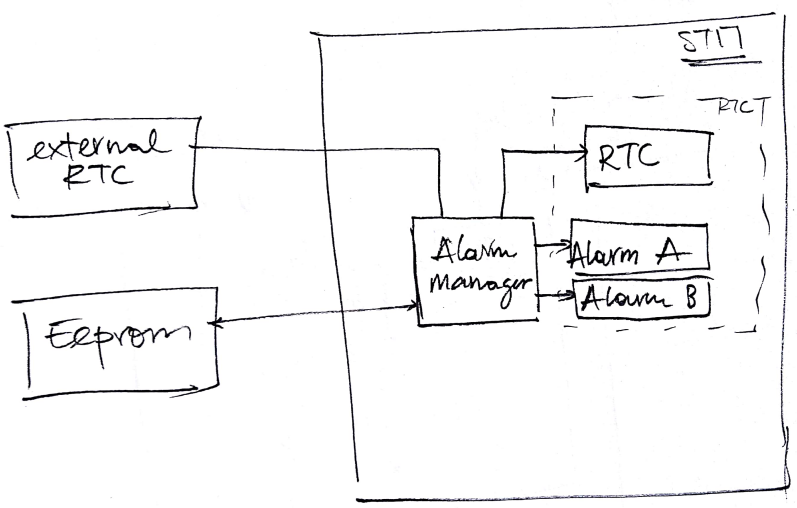


Figure 5.4: Structural diagram of the hardware and software module used in the alarm-clock application.

on the comparison of their time with the internal RTC of the STM. As for the internal (built-in) RTC, it relies on the crystal oscillator of the STM to update its time. This RTC is power dependent and loses its time when the STM is powered off. To prevent manual update of the time everytime the STM is powered on, an external clock module must be used. For this prototype, the DS1307 RTC is used as the external clock module. This RTC is not very accurate, in many applications, it has lost few minutes of accuracy every day. This prototype is a proof of concept and is focused on the light requirements and not the accuracy of the clock, therefore, the DS1307 will be used. Later versions of the NPSC could use more accuracy clock module such as a GPS or more accurate RTC. On power on, the Alarm manager synchronises the internal clock with the external clock by loading the external RTC's time and date to the internal RTC. It then loads two alarms from the EEPROM which are supposed to be triggered before the others and set these alarms to alarm A and alarm B from the internal RTC. The Alarm manager is also responsible for storing the alarms to the EEPROM and updating then when an alarm goes off.

5.2.5 Visual outputs

The visual outputs are all modules related to the control and emission of light in the system. Figure 5.5 illustrates the interaction between these modules and the STM. There are two types of hardware modules, the first type (type 1) of modules is a series

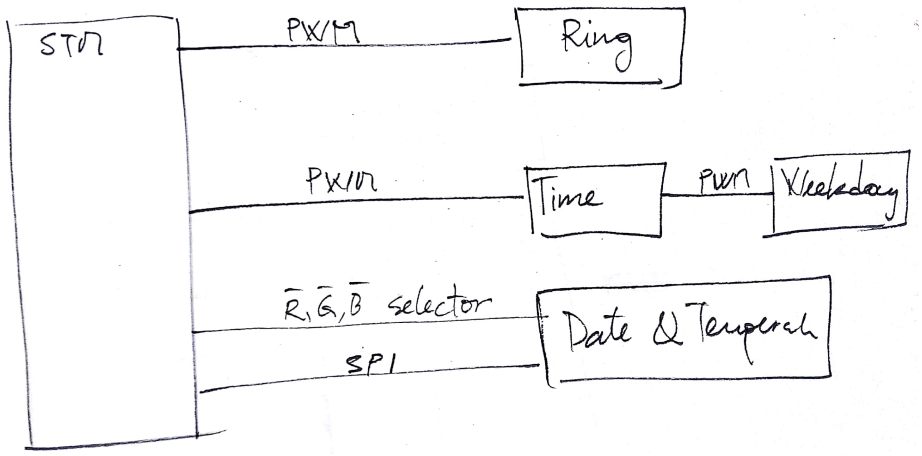


Figure 5.5: Structural diagram of the hardware and software used for the visual outputs.

of neopixels connected in cascade controlled by Pulse Width Modulation (PWM), the second type (type 2) of modules are controlled using a Serial Peripheral Interface (SPI) bus. All type 1 modules inherit the ability to be connected in cascade from the neopixels; however, the ring is the only light source used to affect the human sleep-wake cycle, it is essential that its control is separated from another type 1 module. The two other type 1 is designed to display time and date information, therefore, they can be combined as one module. The only type 2 module in the visual outputs is the Date PCB, this PCB is designed to display the date, the month and the room temperature.
For consistency in the visual, RGB 7 segments displays are used for the Date PCB. Moreover, the Time PCB uses neopixels as no RGB 7 segments displays bigger than the ones used for the Date PCB could be found.
The visual outputs are controlled by the Visual manager from the visual application. The Visual manager's role is to update the neopixel buffer for each type 1 module according to the visual parameters set by the user. These parameters consist of RGB colours, the brightness, the location of the pixel [2]. At a higher level of control, these parameters also include module's pattern, mode and duration. The Visual manager is also responsible for updating information on the Date PCB.

5.2.6 Sensors

There are two sensors on the NPSC, a temperature sensor and a photoresistor. The role of the temperature sensor is to get the room temperature. The photoresistor has a more

[2]The location is the number of the neopixel of interest in the series of neopixel on a specific module. On the Ring PCB, location 61 is the first neopixel on the second ring.

interesting purpose; it is used to obtain the light intensity outside the NPSC in order to allow the system to adjust the neopixel brightness such that the required illuminance does not vary by a significant margin at the user's location.

5.3 Details hardware design

This section provides detailed information on the design of the hardware modules. Off the shelves, hardware modules are not detailed in this sections as these modules already have the necessary information on their datasheets. All datasheets are in B and in the repository on GitHub.

5.3.1 Neopixels boards

The type 1 modules are made up of series neopixels connected in a daisy chain fashion. Each neopixel has two power pins and two data pins as illustrated by fig. 5.6. The stream of bytes is received by the neopixel at its DIN pin, after extracting the first 24 bytes, the neopixel transmit the truncated stream to the DOUT pin. By connecting the DOUT pin of one neopixel to the DIN pin of the next neopixel, a single stream of bytes can be used to program all neopixels in the series. Although all type 1 module uses the

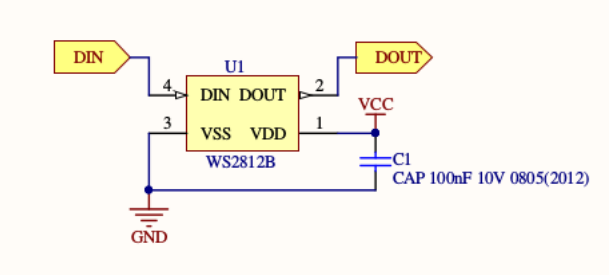


Figure 5.6: Schematics of one neopixels receiving data from DIN and transmitting data to DOUT.

sample principle, each module has more specific features in its design.

Ring PCB

This module has 180 neopixels physical positioned such that they form three rings of 60 neopixels each. Unlike the neopixel in fig. 5.6, there is one capacitor per five neopixels. The daisy chain is broken at every 15^{th} neopixels for debugging the data stream.
As mentioned in table 2.3, each neopixel requires $20mA$ per colour. The ring uses 180 neopixels thus a current of $20 * 3 * 180 = 10800mA$ is expected to be drawn by the Ring. Following the IPC2221 standard, the board width was calculated with a current of $12A$ (worst case scenario), a board thickness of $1oz/ft^2$ and a temperature rise of 15^oC; the required trace width of the external layers must be at least**9.84mm**. This is a stringent requirement on the NPSC as the ring must be contained in a $25 * 25 * 10cm^3$

case. With an expected an inner radius of $6.5cm$ and an outer radius of $10.5cm$, the Ring has approximately a surface area of $\pi * (10.5^2 - 6.5^2) = 213.63cm^2$ on each side for dissipating its heat. Furthermore, the neopixel pins have a pad on $0.6 * 1.5mm^2$ while the width of the power line must be at least $9.84mm$. Each ring has 60 neopixels thus $60 * 9.84 =$ **590.4mm** is required as the width of the power line per ring. However, the minimum circumference of the ring is $65 * 2 * \pi =$ **408.41mm** which is less than what is required. With this conditions, the board is expected to have a higher temperature rise.
The schematic and PCB of this board are shown in fig. A.1.

Time

This board has its series of neopixels physically placed on the board such that they simulate four RGB seven segment display. The board is made generic so that each display can be controlled by its own data pin. The board also has the ability to be controlled all display using the first display controller pin, this is done by connecting the DOUT of the last neopixel of a display to the DIN of the first neopixel in the next display. The schematic and PCB of this board are shown in fig. A.2.

Weekday

This board has only seven neopixels in its series, one for each day in the week. It gets its input (DIN) from the output (DOUT) of the Time PCB. The schematic and PCB of this board are shown in fig. A.3.

5.3.2 Date and Temperature PCB

This PCB uses six seven segments display to display the date, the month and the room temperature. Figure 5.7 shows the simplified schematic of this module. The seven segment used is the ACD8143 which contain a pair of RGB seven segments display. In order to reduce the number of pins used to control these displays, the serial interfaced 8-digits LED driver display (MAX7219) was used. It uses SPI to control the segments of all the displays. It controls the displays by continuously selecting each display in a loop and setting the segments to the desired voltage. The MAX7219 is not designed for RGB seven segments displays, therefore, additional circuits composed of R, G, B signals and NOT gates were used such that on display selection, the display has the colour set by the R, G, B selectors.

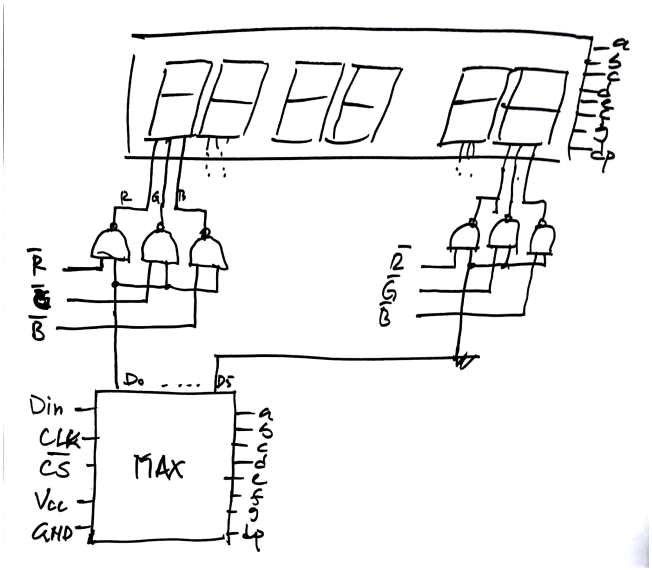


Figure 5.7: Simplified schematic of the Date and Temperature circuit.

Chapter 6

Implementation

This section provides information on the software implementation of the NPSC. Referring to the system hierarchy illustrated in fig. 3.2; this section focuses on the Utilities at level 1, the Framework at level 2, the Applications at level 3 and the main at level 5.
The information on the Utilities and Framework module are given in tabular form the functions in these modules have little complexity and scope [1].

6.1 Utilities

This module contains all functions used by other modules. Table 6.1 describes the use of each file in the Utilities module.

Table 6.1: Description of the files at the Utilities level.

File Name	Description
queue	Contains functions to create a queue of a certain data type, add and remove elements from the queue.
utils	Contains functions, variables and type definitions used other software modules at different levels of the hierarchy.

6.2 Framework

The files in this module are the framework code related directly to each hardware module. Table 6.2 describes the use of each file and their corresponding hardware module.

[1] These functions do not rely on other functions, most of these are directly linked to the hardware or perform simple logic.

Table 6.2: Description of the files at the Framework level.

File Name	Hardware	Description
audio	N/A	Load ringtones from the EEPROM, initialise the DAC peripheral and provide functions to play and stop the ringtone.
bluetooth	HC-06	Initialise the UART communication between the STM and the HC-06 and provide functions to send, receive data.
clock	internal RTC	initialise and activate the internal RTC on the STM and provide functions to set, get the clock as well as the alarms.
eeprom	25LC640	Initialise the SPI communication between the EEPROM and the STM and provide functions to read/write from/to the EEPROM.
neopixel	WS2812	Initialise the STM's PWM on the neopixel peripherals and provide functions to set colours of specific or all neopixels.
nextion	NX4024T032_001	Initialise the UART communication between the STM and the Nextion screen and provide functions to send, receive data.
rtc	DS1307	Initialise the I2C communication between the STM and the external RTC and provide functions to set and get the time and date.
temperature	MCP9700	Initialise the ADC peripheral on the STM and provide functions to read the temperature.
ldr	LDR	Initialise the ADC peripheral on the STM and provide functions to read the light intensity.
framework	All	Initialise all hardware communication

6.3 Applpication

Unlike the module described in the Utilities and Framework modules, the applications described in this section use multiple modules at the application, framework and utilities level.

6.3.1 Visual application

The visual application is the most active application in the system. It refreshes the visual output at a user-defined rate which is by default *1s*. Figure 6.1 shows the flow of the timer process. After the timer is set, certain variables indicator of a timer trigger event are set. These timer indicator are later used by the *visual update* process illustrated

in fig. 6.2. The *visual update* process starts by verifying that the timer indicator of

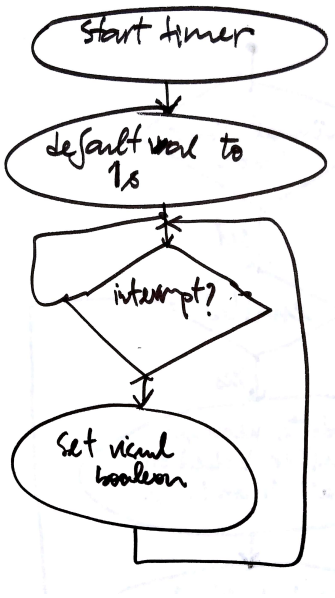


Figure 6.1: Flow diagram of the visual timer process.

the specific visual module is set. If so, it then checks that the corresponding visual module is enabled. If the module is enabled, the neopixel buffer of the corresponding module is updated based on the pattern selected and an index indicating what should be the next iteration of the pattern. The process continues by display the pattern on the corresponding module and resetting the timer indicator.

6.3.2 Alarm application

This application has one main process. This process consists of calling another process (*alarm_update*) when the device is powered on or when an alarm has been updated in the EEPROM or has been triggered. Figure 6.3 gives the flow of this process. The *alarm_update* process loads the two alarms which will be triggered before the others, into Alarm A and B from the internal RTC. To make this decision, the alarms in the EEPROM are compared by converting the alarm into an integer value. The lower the value, the sooner the alarm will be triggered.

6.3.3 Instruction application

The operations performed in the instruction application can be divided into three principal operations, the *DMA interrupt*, the *device instruction update*, and the *instruction fetch decode execute*.

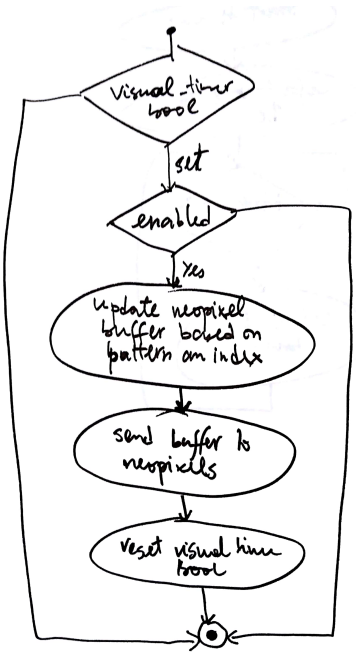


Figure 6.2: Flow diagram of the visual update process. Each visual output module uses the same flow to update their visual.

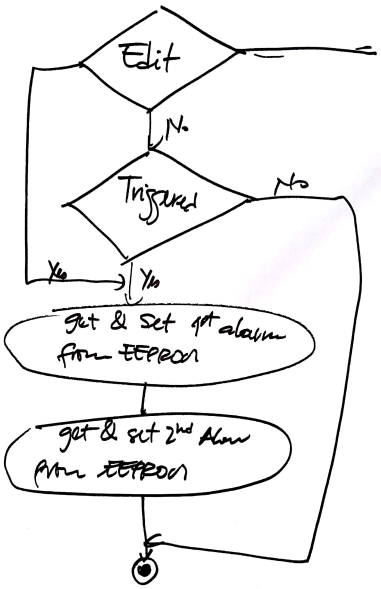


Figure 6.3: Flow diagram of the alarm update process.

DMA interrupt

There are two user input devices each using the DMA and a DMA buffer to store the byte sent to the STM. Each DMA generates an interrupt on data reception which is handled by the corresponding interrupt handler. When the device is powered on, the CPU initialise the communication with the input device, then it continuously observes the DMA and generates an interrupt when data is received. When this occurs, the DMA interrupt handler then set a boolean value to indicate that data was received from the

input device. The flow of the process described above is illustrated in fig. 6.4.

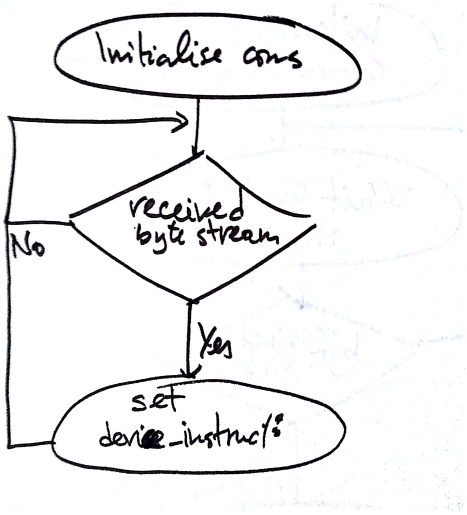


Figure 6.4: Flow diagram of the DMA process.

Device instruction update

The purpose of this process is to convert the data stored in the DMA Buffer into instructions and add these instructions to the instruction buffer. When the process is called, the boolean set by the DMA interrupt process is evaluated. When its value is *reset*, it implies that the DMA buffer did not receive any new instructions. If the boolean was set, the DMA buffer is divided into sections of eight bytes each (size of an instruction) and each section is converted into instructions, then added to the instruction queue and the boolean is reset. Prior the conversion of the DMA buffer section to instructions, the data contained in these sections are analysed and sections which do not match any instruction type are discarded. The flow of this process is illustrated in fig. 6.5.

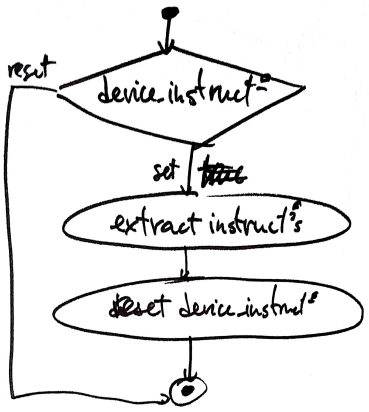


Figure 6.5: Flow diagram of the input device instruction update. Both Nextion screen and Bluetooth application uses the same flow to update the instruction queue.

Instruction Fectch Decode Execute (IFDE)

The IFDE's purpose is to fetch, decode and execute the instructions in the instruction queue. This process starts by checking that there are instructions in the instruction queue. It then fetches these instructions one by one, decode and execute the instructions. The process stops when the instruction queue is empty. This process is illustrated in fig. 6.6 and a portion of the code used for its implementation is shown in listing 6.1. .

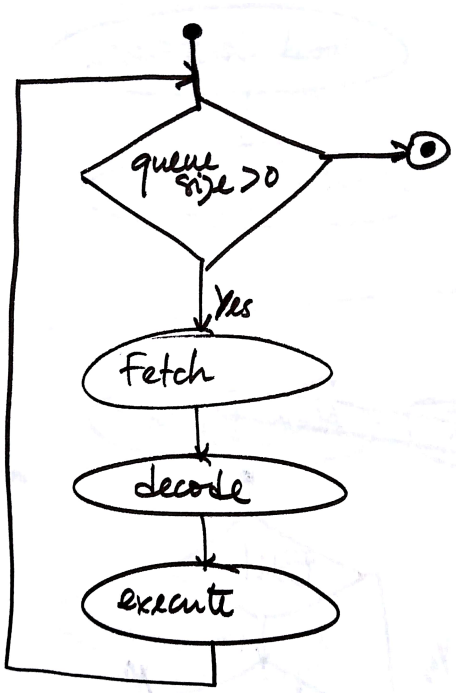


Figure 6.6: Flow diagram of the IFDE process.

```
/**
 * @brief    Execute instructions from instruction queue
 */
void instruction_execute(void){
    RTC_ClockTypeDef clock;
    AlarmTypeDef alarm;
    // check if the queue is empty
    while(InstructionQueue_isEmpty(instruction_queue)){
        // Fecth instruction
        InstructionTypeDef _instruction = InstructionQueue_dequeue(
    instruction_queue);
        // Decode and Execute instruction
        switch(_instruction.instrution[0]){
        /* External RTC instructions */
        case 0x01:
            // set clock
            clock.date.RTC_Year=_instruction.instrution[1];
            clock.date.RTC_Month=_instruction.instrution[2];
            ...
            rtc_setClockStruct(&clock);
            break;
```

```
21          case 0x02:
22              // get clock
23              instruction_nextionStart();
24              ...
25              instruction_nextionSendStr("home.t2.txt=", rtc_monthToString(
    clock.date.RTC_Month) );
26              instruction_nextionSendInt("home.n4.val=",2000+clock.date.
    RTC_Year);
27              instruction_nextionStop();
28                break;
29          ...
30          }
31      }
32  }
```

Listing 6.1: Instruction Fetch Decode Execute

After the instruction *_instruction* is extracted from the *instruction_queue*, the first byte (*_instruction[0]*) is used to decode the instruction based on the instruction set defined in table 5.1. There are two types of instructions, the *get instructions* and the *set instructions*. The *set instructions* contains parameters used to update the hardware modules of the NPSC. The *get instructions* are used to signal the NPSC that the input device required some parameters update. Each device response to the *get instructions* is different. The Bluetooth response is not complex and consists on sending an instruction (eight bytes) to the Bluetooth device. The Nextion touchscreen response is more complicated; after sending a *get instruction* to the STM, the Nextion touchscreen waits for instruction. The STM replies to this request by starting a three phases protocol consisting of:

- **instruction_nextionStart()**: The STM sends random data to signal the touchscreen that useful data will arrive.
- **instruction_nextionSend<type>(char *, type)**: Special Nextion commands are used to update the screen parameters.
- **instruction_nextionStop()**: The STM ends the transmission by sending a *"com stop"* instruction to the screen.

6.4 Main

The main or the master application has one unique role: *to start all processes*. This application calls the initialization functions of each level which start the hardware communication protocols, the instruction update process, the IFDE process and so forth.

Chapter 7

Testing and Experimentation

This section provides information on the software and hardware tests of the NPSC. It starts by providing an overview of the model used to test the NPSC. It continues by providing all unit tests and system tests performed not the NPSC. Lastly, it explains the current, temperature and light experiments done on the Ring.

7.1 Overview

Debugging an embedded system is one of the worst things in life, especially when there are logical errors in the program. In an embedded system like the NPSC which is made out of modules interacting with each other at different levels of the hierarchy, finding an error would be a difficult task as the error would have to be tracked down to the modules at lower levels. To reduce debugging time, it is essential that each module is independently and thoroughly tested. To achieve this, the verification steps of the v-diagram of the NPSC shown in fig. 7.1 are implemented.
The hardware modules and framework are tested by analyzing the data transmitted between the STM and these hardware modules, and by running a set of unit tests on the framework code. The sub-systems of the NPSC are tested by running a set of software tests. Following the integration tests is the system test which consists of using the device to ensure that the system requirements have been met. The last verification steps consist of verifying that the device is an alarm-clock able to produce light of wavelength around 460nm and able to fit in a $25 * 25 * 10 cm^3$ case.

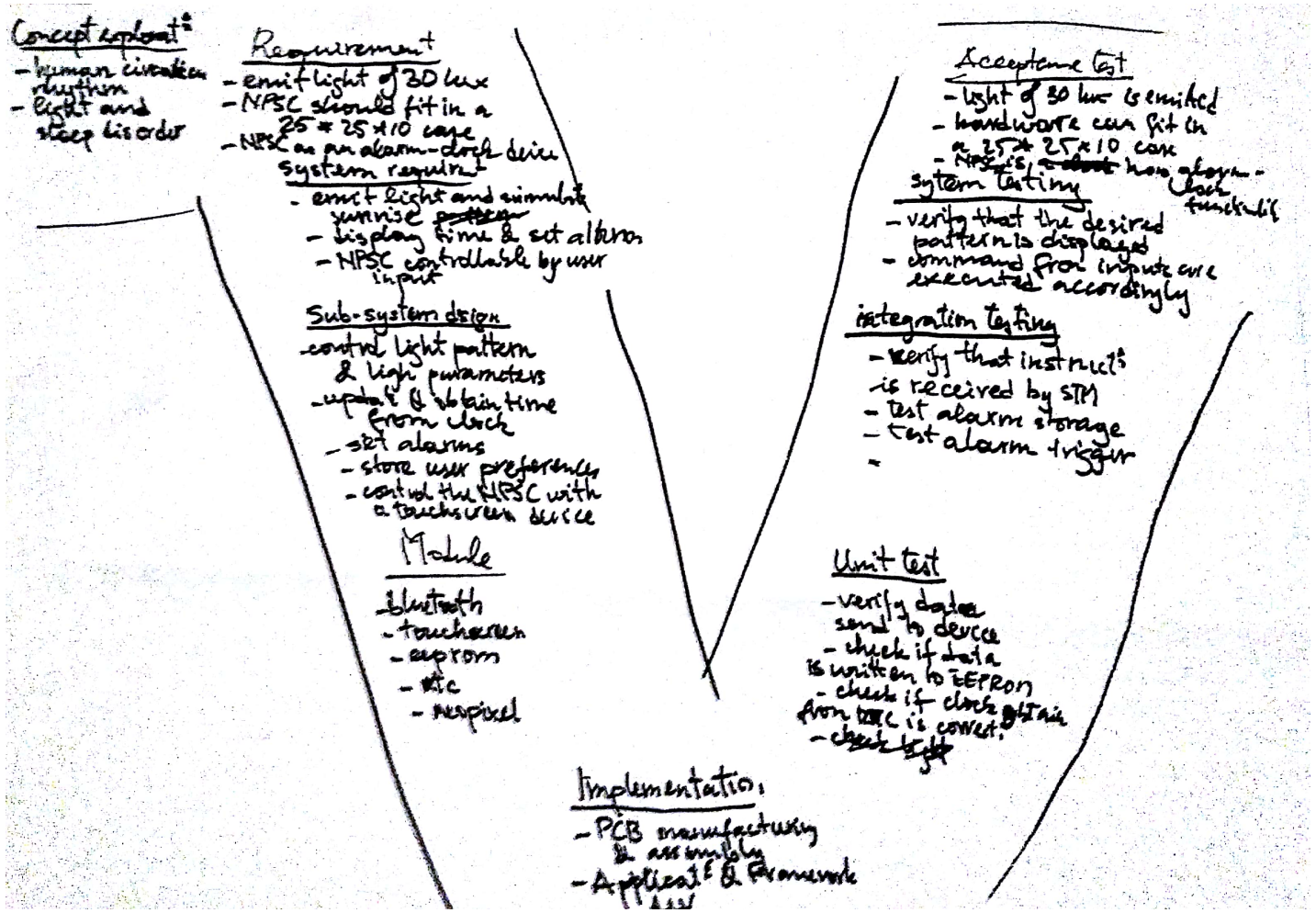


Figure 7.1: V-diagram of the project. On the left side, from top to bottom are the verifications steps starting from the system requirements down to the modules. At the bottom is the implementation of the system On the right side, from the bottom to the top are the validations steps starting from the unit tests to the acceptance tests performed on the NPSC.

7.2 Hardware module tests

The tests performed on the hardware are the basic hardware tests mentioned in 3.5.1. To ensure that the hardware were sending and receiving the right information, the *Salea logic analyser* was used to analyse the data transmitted between the STM and the hardware modules.Table 7.1 describes the communication tests performed on each module.

Table 7.1: Commnication protocol tests performed on hardware modules.

Hardware	**Protocol**	**Test description**
DS1307	I2C	rtc_setMinutes(10). Send the value 10 at address 0x01 of the RTC.
25LC640	SPI	eeprom_write(10,0x00). Send SPI data of write type to write the value 10 at address 0x00.
HC-06	UART	bluetooth_send(10). Send UART data of write type to send the value 10.
NX4024T032_001	UART	nextion_send(10). Send UART data of write type to send the value 10.
MAX7221	SPI	date_write(10,0x00). Send SPI data of write type to write the value 10 at address 0x00.
WS2812	PWM	neopixel_setAllPixelRGB(255,0,0) neopixel_setAllPixelRGB(0,255,0) neopixel_setAllPixelRGB(0,0,255). Send PWM data to the neopixels.

7.3 Unit tests

Software unit tests could not be run on all hardware modules. It would have been impractical to set software unit tests on the Bluetooth module and the touchscreen as an application need to be developed using those modules before fully interacting with them. The Utilities modules and the Framework of the hardware modules tested using unit testing are listed in table 7.2 alongside their unit tests.

7.4 Integration tests

The tests performed on the applications are more complex than the unit tests. It is important that the unit tests are run prior this tests as the applications rely on the framework. The description of all integration tests performed on the applications is shown in table 7.3.

Table 7.2: Unit tests performed on the Framework.

Hardware	Unit test	Description
internal RTC	test_clock_date	Assert that the date saved is the same as the date loaded from the internal RTC
	test_clock_time	Assert that the time saved is the same as the time loaded from the internal RTC
	test_clock_alarm	Assert that the alarm saved is the same as the alarm loaded from the internal RTC
external RTC	test_rtc_clock	Assert that the clock written to the external RTC is the same as the one loaded from the external RTC
queue	test_queue_create	Assert that a queue of specific datatype has been created
	test_queue_enqueue	Assert that an element has been successfully added to the queue
	test_queue_dequeue	Assert that an element has been removed from the queue
eeprom	test_eeprom_write_read	Assert that the byte written to the eeprom is the same as the byte read at the written address.
	test_eeprom_write4B_read4B	Assert that the a uint32_t written to the eeprom is the same as the uint32_t read at the written address.
	test_eeprom_writeNB_readNB	Assert that the N bytes written to the eeprom are the same as the N bytes read at the written address.

Table 7.3: Integration test performed at the application level.

Application	Integration test	Description
alarm	test_alarm_address	Assert that the address of all type of the Alarm structure are offset by the correct memory size.
	test_alarm_save_load	Assert that the alarm save in the EEPROM is the one loaded from the EEPROM

7.5 Neopixel Ring

The Ring was designed so that it emits light with a wavelength of $460 \pm 10nm$ and at an illuminance of $30lux$ while being able to draw a current of $12A$ with a temperature rise of 15^oC. Three experiments were conducted on the board to quantify its current consumption, its temperature rise and its illuminance.

7.5.1 Current experiment

This experiment was done to verify that the board can draw $12A$ without blowing up (not kidding) and to find a mathematical model between the neopixel brightness and the board current consumption.
The first part of the consisted of finding the current drawn by a single neopixel based on its colour and brightness. The brightness levels are from 0% to 100% in steps of 10%, so there were 11 brightness levels in total. The 0% brightness level is defined as the idle state of the neopixel as all pixels are not turned on.
The second part of this test consisted of measuring the current consumption of the Ring with all 180 neopixels. Firstly the idle current of the board was measure (current when the board is powered and no data is sent). Secondly, the current per brightness level was measured.
In this experiment and all described below, the *TOPWARD, DC power supply 33010D* capable of supplying a current of $10A$ was used.

Figure 7.2: Power supply used in the Ring experiments.

7.5.2 Temperature experiment

The temperature experiment was performed in order to find the temperature rise of the board per brightness level. In this experiment, all 180 neopixels were turned white (100% red, 100% green,100% blue).
Four temperature sensors (MCP9700) were evenly placed at the back of the board as illsutrated by fig. 7.3 to measure the board temperatue. Another temperature sensor was placed further away from the Ring to measure the room temperature. An Arduino Uno was used to run this experiment by controlling the neopixels brightness and colour

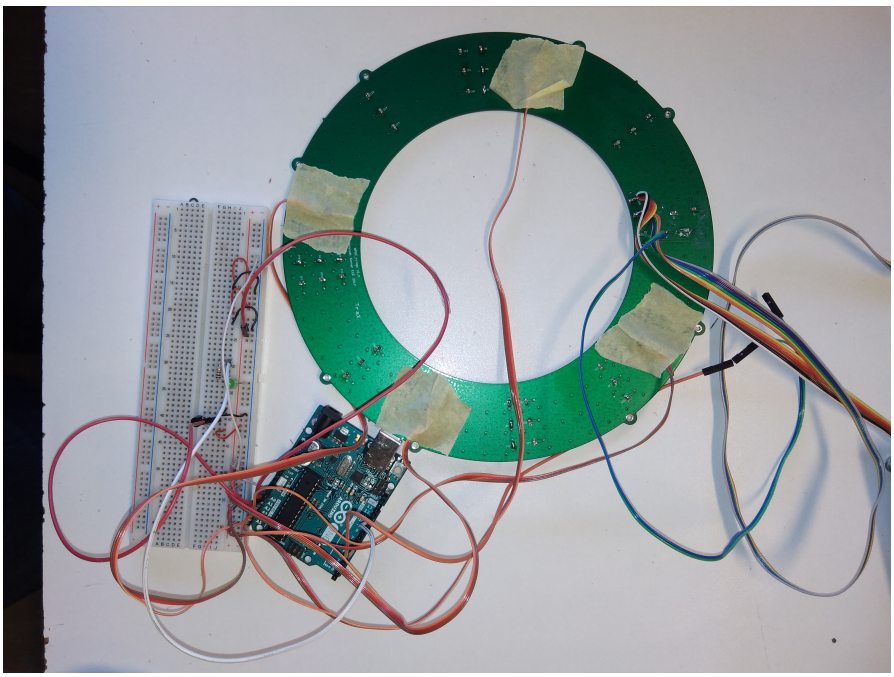


Figure 7.3: Schematics used to obtain the board temperature.

and by controlling the temperature sensors. The schematic in fig. 7.4 was used to set up the experiment. In this schematic, T_0 is the temperature sensor measuring the room temperature and the temperature sensors T_1, T_2, T_3, and T_4 are the sensors placed on the board. Different power supplies were used in this experiment, the $5V$ @ $10A$ power supply was used to power the Ring while a $5V$ @ $0.5A$ power supply coming from the usb port of a laptop was used to power the Arduino and the temperature sensor. An LED was used to display the outcome of the experiment. Two programs were written for this experiment. The first program compares the average temperature given by sensors on the board with the room temperature and turns on the LED if those temperatures are the same. The second program averages the temperature of the sensors on the boards and display the sample number, the board temperature and the room temperature at interval of $10s$ on the Arduino's serial monitor.
The temperature of the board measure for each brightness level using all 180 neopixels with white colour. After each temperature measurement, the sensors were let to cool down to the room temperature using the first program written. The results of the experiment were taken from the serial monitor.
The overall setup of this experiment is shown in 7.5.

7.5.3 Illuminance experiment

This experiment was performed to quantify the illuminance of the Ring in order to verify if it meets its requirements. Moreover, the experiment's purpose was also to find a relationship between the distance and illuminance, the relationship between the angular position of the subject and the illuminance.

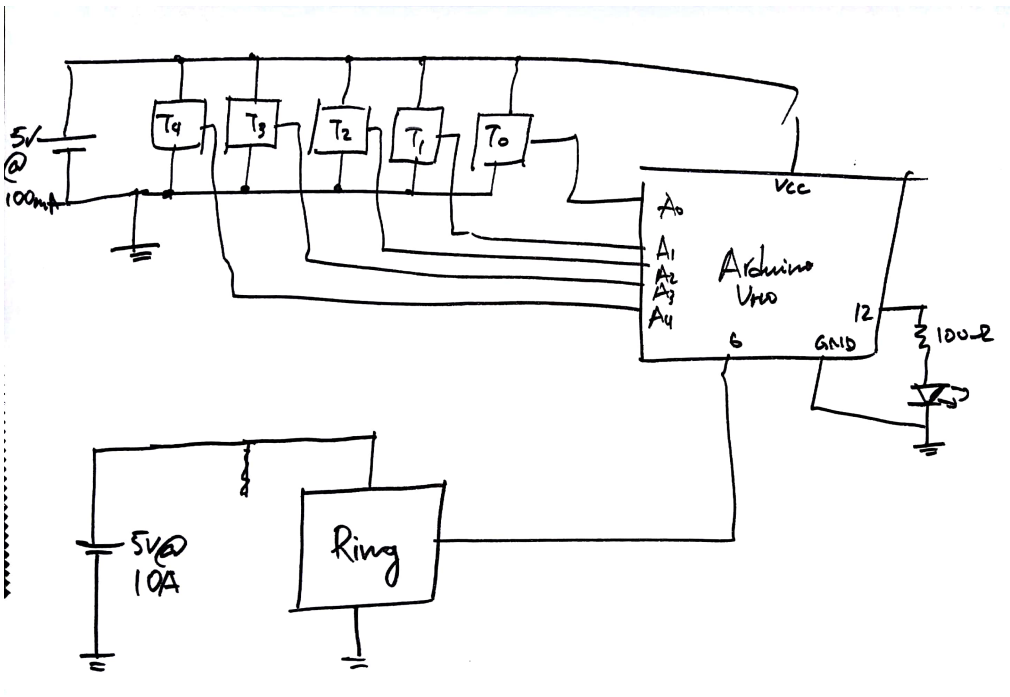


Figure 7.4: Schematics used to obtain the board temperature.

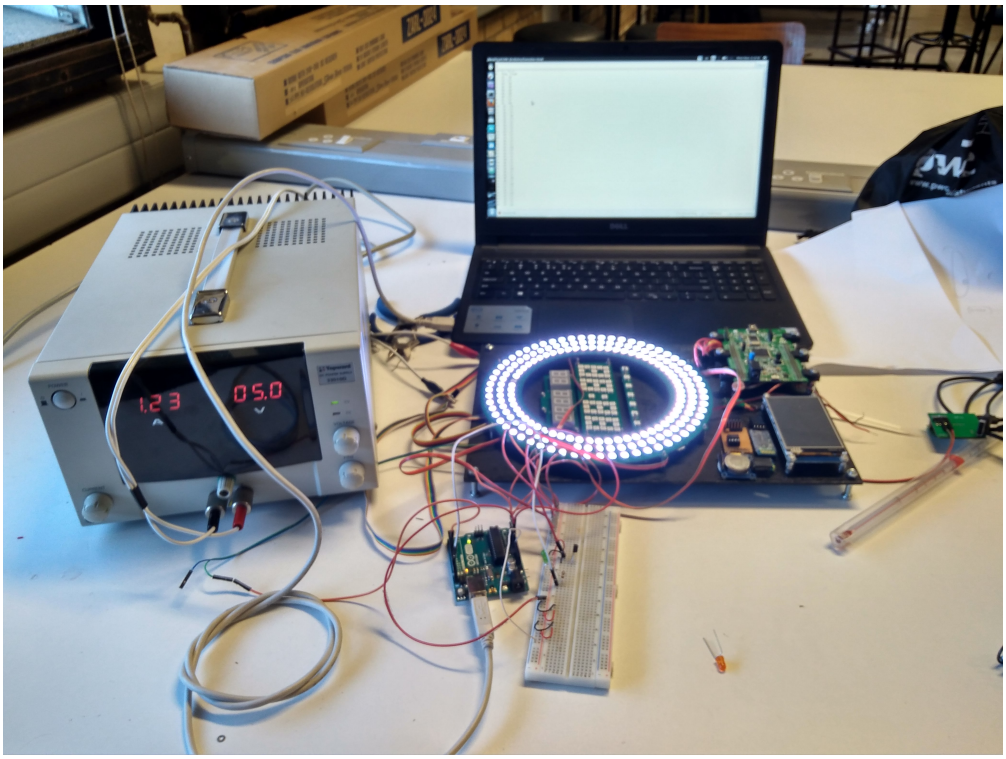


Figure 7.5: Setup of the temperature rise experiment.

To ensure that the Ring's illuminance is properly measured, the experiment was performed in room EM6 in the Menzie Building at the University of Cape Town. The room's lights were turned off and all windows were closed so that a reading of $0lx$ was obtained on the lux meter. The lux meter used in the experiment is the *LTR55X ALSPRX*. It has a

resolution of $0.01499930lx$ and a maximum range of $10000.0lx$. This lux meter is found in the *Xiami Redmi 4* smartphone, it was used over other digital lux meters as there were not any available. For this reasons, the wavelength emitted by each LED of the neopixel could no be verified; only the lux emitted by the neopixel could be quantified. The illuminance was measured for distances from the Ring ranging from $10cm$ to $100cm$ with steps of $10cm$. The experiment was performed at 0^o, 30^o, 60^o, and 90^o angle, with the angle being measured from the normal to the front of the Ring. The Ring was placed on a table and the lines indicating the angles and the distance to the Ring was drawn on the same type of tables. Figure 7.6 shows the positions of the Ring for the experiments. After setting up the environment for the experiment, the illuminance of

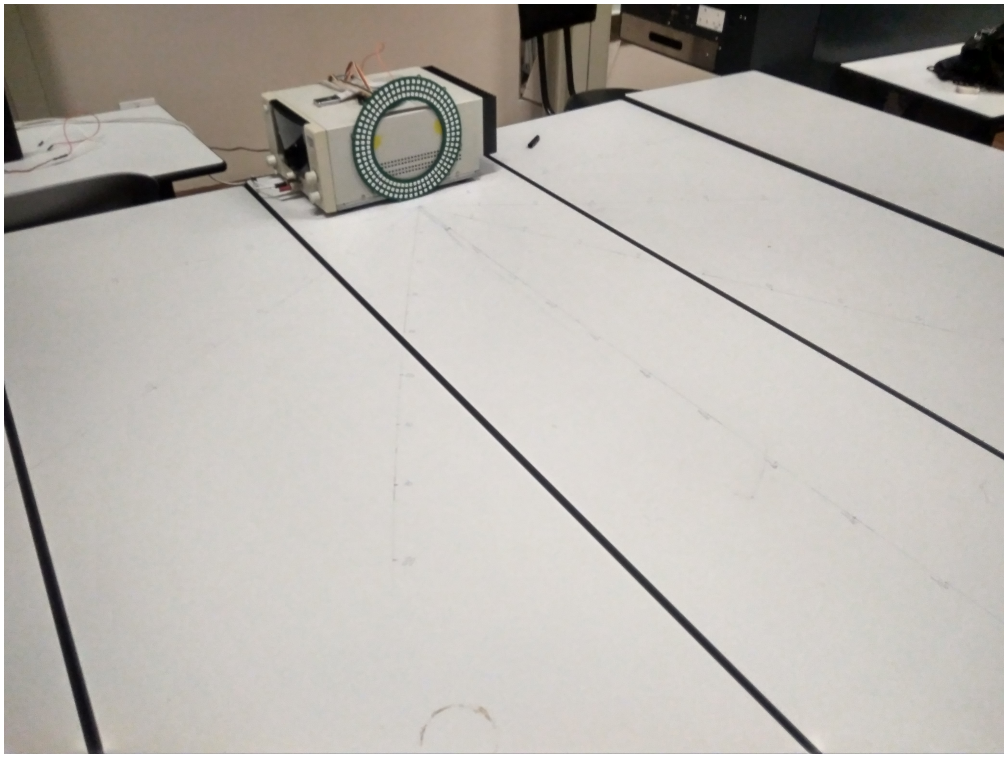


Figure 7.6: Illuminance test setup.

the Ring was taken at brightness level of 10%, 30%, 50%,80%, and 100%. For each brightness the illuminance was measure at 0^o, 30^o, 60^o, and 90^o angle. And for each angle, the brightness was measure at distance of $10cm$ to $100cm$ with steps of $10cm$.

Chapter 8

Results and Discussions

8.1 Software Tests

8.1.1 Tests outcome

Table 8.1: Unit and Integration test outcome.

Hardware	Unit test	Outcome	Coverage estimation
internal RTC	test_clock_date	Pass	33.33%
	test_clock_time	Pass	
	test_clock_alarm	Pass	
external RTC	test_rtc_clock	Pass	47.1%
queue	test_queue_create	Pass	44.44%
	test_queue_enqueue	Pass	
	test_queue_dequeue	Pass	
eeprom	test_eeprom_write_read	Pass	66.67%
	test_eeprom_write4B_read4B	Pass	
	test_eeprom_writeNB_readNB	Pass	
alarm	test_alarm_address	Pass	66.67%
	test_alarm_save_load	Pass	

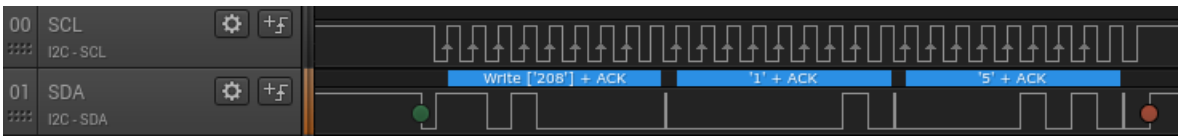
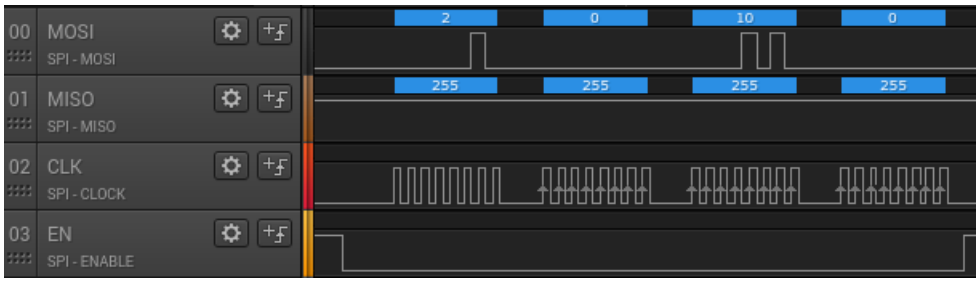


Figure 8.1

(a)



(b)

Figure 8.2

8.1.2 Insight on the software tests

8.2 Hardware module tests

8.2.1 Test results

DS1307

25LC640

HC-06

Figure 8.3

Figure 8.4

NX4024T032_001

WS2812

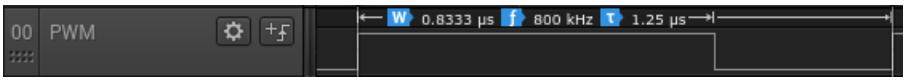


Figure 8.5

(a)

(b)

(c)

Figure 8.6

Table 8.2: Current drawn by a one neopixel per colour at different brightness levels. The white colour is obtain by turning the red, green and blue LEDs on at the same brightness.

Brightness (%)	**Current (mA)**			
	Red	**Green**	**Blue**	**White**
0	3	3	3	3
10	4	4	4	5
20	5	5	5	9
30	7	7	6	13
40	8	8	8	17
50	10	10	9	22
60	11	11	11	26
70	13	12	12	30
80	14	14	13	34
90	15	15	15	38
100	17	17	16	42

Table 8.3: Current drawn by the neopixels in idle mode. The idle mode is defined as the state of the of the neopixel when no light is emitted.

Number of neopixels	**Current (mA)**
0	89
10	89
25	90
50	92
90	95
120	97
150	99
180	101

Table 8.4: Current drawn by all 180 neopixels on the Ring at different brightness levels.

Brightness (%)	**Current (A)**			
	Readings			**Average**
0	0.120	0.120	0.120	0.120
10	0.510	0.510	0.500	0.507
20	1.250	1.230	1.220	1.233
30	2.000	1.980	1.970	1.983
40	2.730	2.690	2.680	2.700
50	3.480	3.430	3.410	3.440
60	4.890	4.120	4.110	4.373
70	4.910	4.840	4.830	4.860
80	5.600	5.530	5.510	5.547
90	6.320	6.240	6.220	6.260
100	7.020	6.910	6.890	6.940

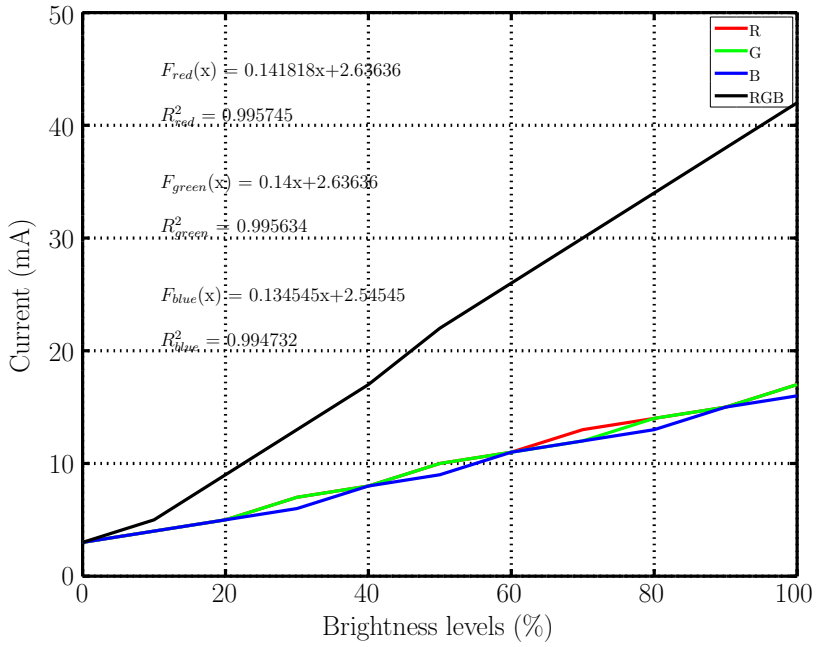


Figure 8.7

Table 8.5: Current drawn by the neopixels in idle mode. The idle mode is defined as the state of the of the neopixel when no light is emitted.

Current (A)	**Temperature (^oC)**
0.09	2.176
0.49	6.417
1.24	11.750
1.95	17.667
3.38	28.083
4.1	29.667
5.35	39.000
6	43.091
6.55	44.750

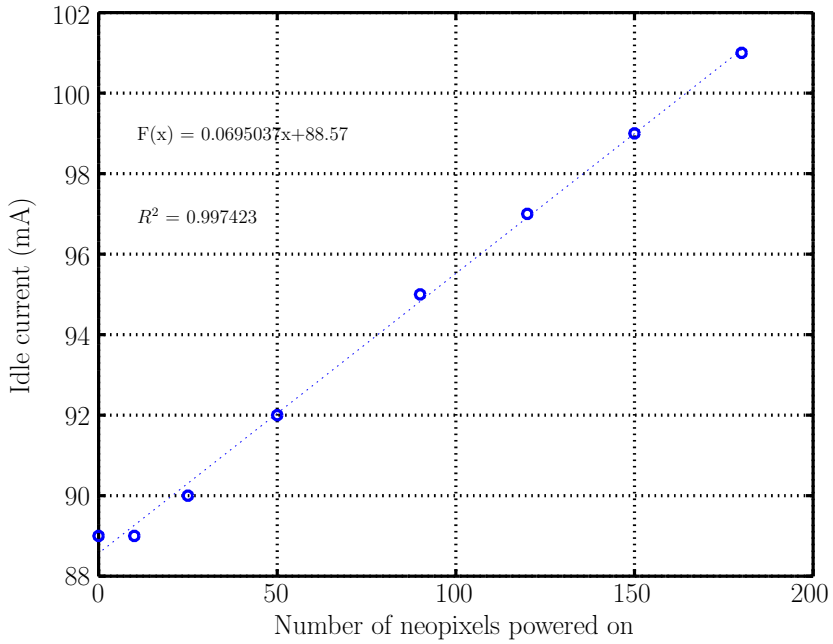


Figure 8.8

Table 8.6: Ring's blue LED illuminance at full brightness per distance and angular section to the Ring.

Distance (cm)	**Angle (deg)**			
	0	30	60	90
20	6420	5950	5310	1998
30	4910	3770	3302	547
40	2507	2354	1657	149
50	1820	1620	1219	110
60	1375	1227	892	60
70	1012	943	691	55
80	810	785	547	43
90	667	607	420	35
100	557	500	353	30

8.2.2 Discussions

8.3 Ring experiments

8.3.1 Current drawn by the Ring

Experimental results

Discussion

8.3.2 Ring temperature by the Ring

Experimental results

Discussion

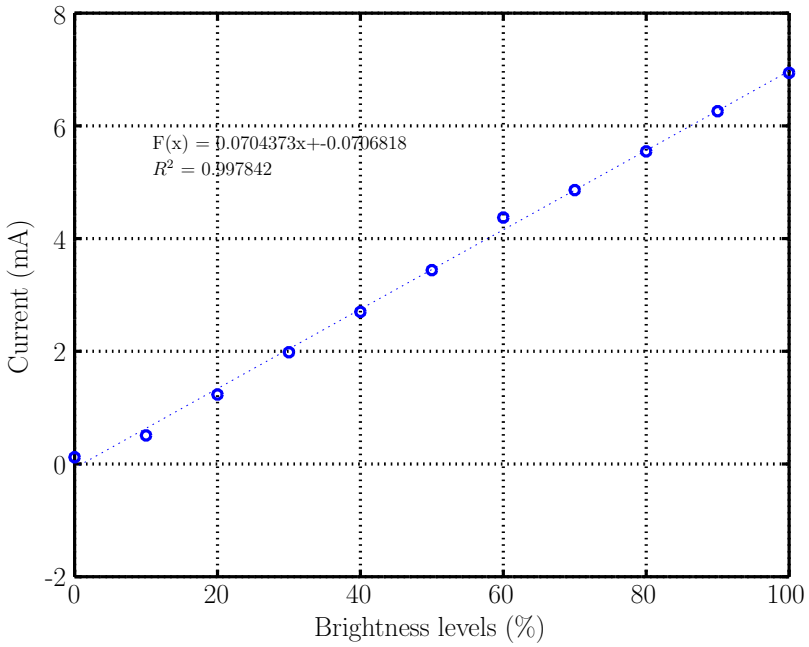


Figure 8.9

Table 8.7: Relationship between the coefficient of decadence of the Ring illuminance and the distance to the Ring.

Distance (cm)	**Brightness (%)**					**Average**	**Std dev**
	10	30	50	80	90		
20	1.00	1.00	1.00	1.00	1.00	1.00	0.00
30	0.68	0.63	0.65	0.65	0.59	0.64	0.03
40	0.44	0.40	0.44	0.41	0.36	0.41	0.03
50	0.31	0.29	0.31	0.28	0.27	0.29	0.02
60	0.23	0.20	0.23	0.22	0.20	0.22	0.01
70	0.18	0.15	0.18	0.17	0.15	0.17	0.01
80	0.14	0.13	0.14	0.13	0.12	0.13	0.01
90	0.12	0.10	0.11	0.13	0.10	0.11	0.01
100	0.10	0.08	0.09	0.09	0.08	0.09	0.01

Table 8.8: Relationship between the Ring illuminance and the angle to the normal of the Ring surface.

Angle (cm)	**Distance (cm)**					**Average**	**Std dev**
	20	30	50	80	100		
0	1.00	1.00	1.00	1.00	1.00	1.00	0.00
30	0.88	0.89	0.53	0.94	0.90	0.83	0.16
60	0.75	0.70	0.65	0.64	0.64	0.68	0.05
90	0.20	0.16	0.09	0.06	0.05	0.11	0.07

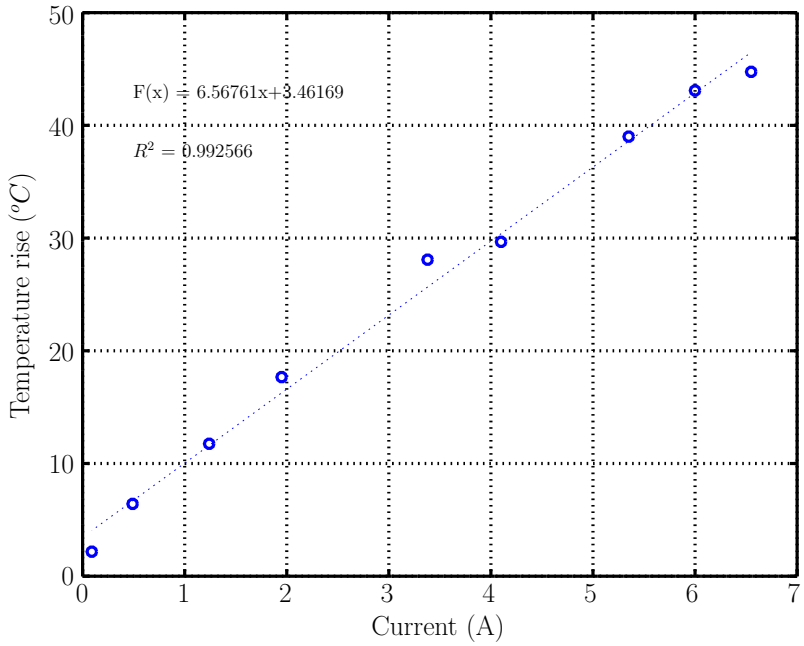


Figure 8.10

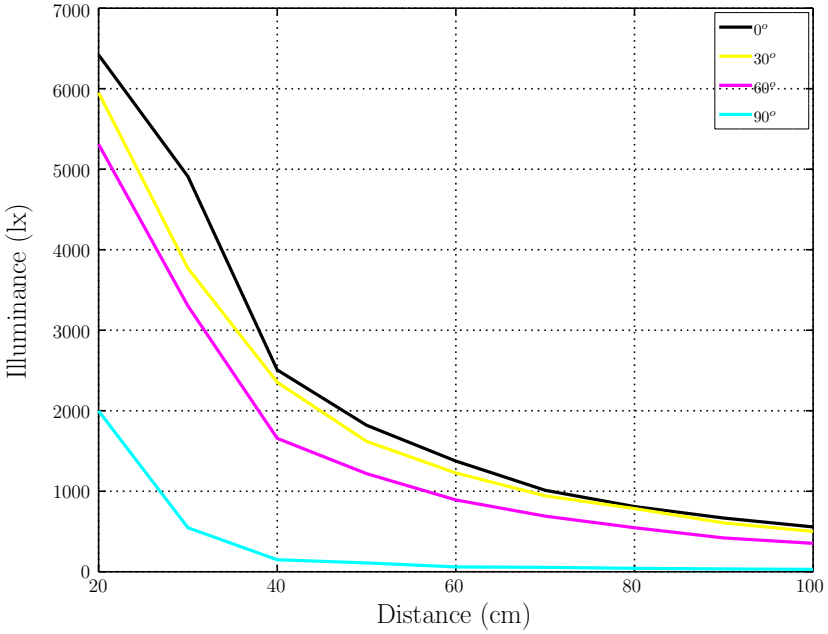


Figure 8.11

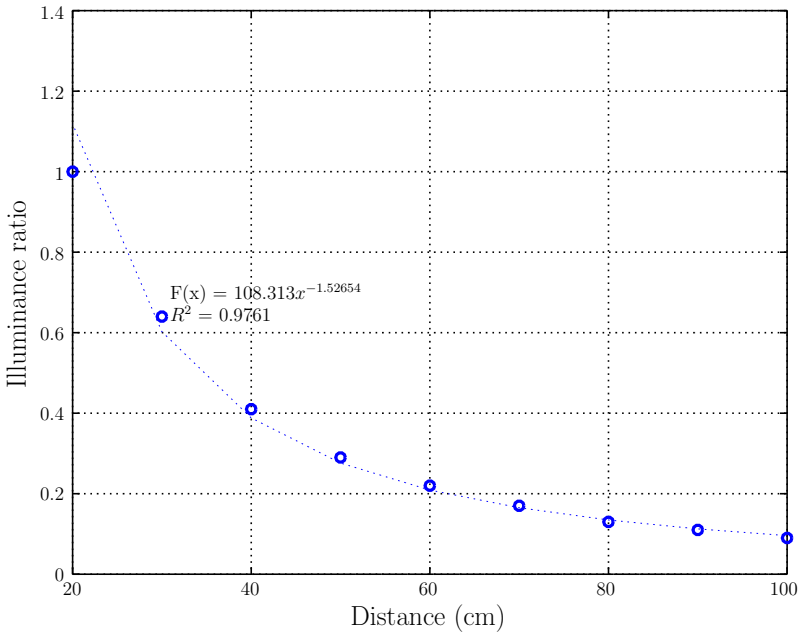


Figure 8.12

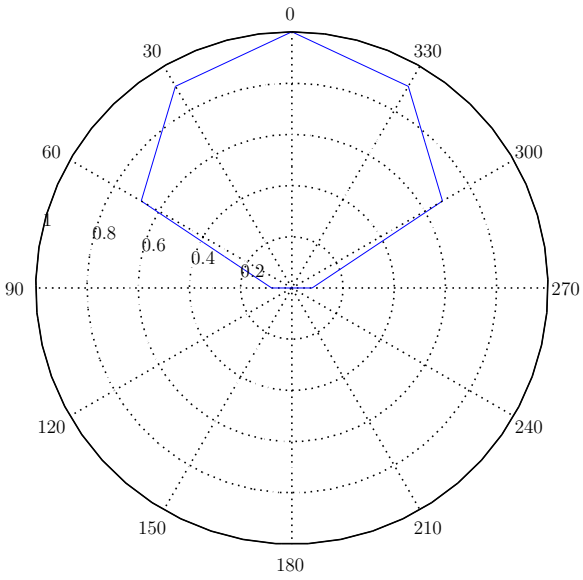


Figure 8.13

Chapter 9

Conclusions

These are the conclusions from the investivation and how the investigation changes things in this field or contributes to current knowledge...

Draw suitable and intelligent conclusions from your results and subsequent discussion.

Chapter 10

Recommendations

Make sensible recommendations for further work.

Use the IEEE numbered reference style for referencing your work as shown in your thesis guidelines. Please remember that the majority of your referenced work should be from journal articles, technical reports and books not online sources such as Wikipedia.

Bibliography

[1] General Electric Company, "GE Lighting, lighting and sleep", December 2014.

[2] C. Cajochen, S. L. Chellappa and C. Schmidt, "Circadian and Light Effects on Human Sleepiness-Alertness", *Sleepiness and Human Impact Assessment*, pp. 9-22, 2014.

[3] Gregory M. Brown, "Light, Melatonin and Sleep-Wake Cycle", *Journal Psychriatry Neurosci*, **vol. 19(5)**, pp. 345-353, Nov 1994.

[4] K. E. West, M. R. Jablonski, B. Warfield, K. S. Cecil, M. James, M. A. Ayers, J. Maida, C. Bowen, D. H. Sliney, M. D. Rollag, J. P. Hanifn and G. C. Brainard, "Blue light from light-emitting diodes elicits a dose-dependent suppression of melatonin in humans", *Journal Appl Physiol*, **vol. 110**, pp. 619-626, 16 Dec 2010.

[5] George C. Brainard, John P. Hanifin, Jeffrey M. Greeson, Brenda Byrne, Gena Glickman, Edward Gerner and Mark D. Rollag, "Action Spectrum for Melatonin Regulation in Humans: Evidence for a Novel Circadian Photoreceptor", *Journal of Neuroscience*, **vol. 21(16)**, pp. 6405-6412, 15 Au 2001.

[6] Berson, D. M., F. A. Dunn, and M. Takao. "Phototransduction by Retinal Ganglion Cells That Set the Circadian Clock." *Science*, **vol. 295**, pp. 1070-073, 2002.

[7] Kavita Thapan, Josephine Arendt and Debra J. Skene, "An action spectrum for melatonin suppression: evidence for a novel non-rod, non-cone photoreceptor system in humans", *Journal of Physiology*, **vol. 535.1**, pp.261-267, July 2001.

[8] Christian Cajochen*, Jamie M. Zeitzer, Charles A. Czeisler, Derk-Jan Dijk, "Dose-response relationship for light intensity and ocular and electroencephalographic correlates of human alertness", *Behavioural Brain Research* , **vol. 115**, pp.75-83, May 2000.

[9] Helen J. Burgess and Charmane I. Eastman, "Early versus late bedtimes phase shift the human dim light melatonin rhythm despite a fixed morning lights on time", *Neurosci Lett.*, **vol. 356(2)**, pp. 115118, Feb 2004.

[10] Helen J. Burgess and Thomas A. Molina, "Home Lighting Before Usual Bedtime Impacts Circadian Timing: A Field Study", *Photochem Photobiol.*, **vol. 90(3)**, pp. 723726, 2014.

[11] US Department of Energy, "Lighting for Health: LEDs in the New Age of Illumination", *Solid-State Lighting Technology Fact Sheet* , 2014.

[12] Mark S. Rea and Mariana G. Figueiro, "A Working Threshold for Acute Nocturnal Melatonin Suppression from WhiteLight Sources used in Architectural Applications", *Lighting Research Center, Rensselaer Polytechnic Institute, Troy, New York, USA*, 2013.

[13] Derk-Jan Dijk, "Internal rhythms in humans", *CELL & DEVELOPMENTAL BIOLOGY*, **vol. 7**, pp. 831-836, 1996.

[14] "Scientific Background Discoveries of Molecular Mechanisms Controlling the Circadian Rhythm", *The Nobel Assembly at Karolinska Institutet*, 2017.

[15] Tobler I., "Is sleep fundamentally different between mammalian species?", *Behav Brain Res.*, **vol. 69(1-2)**, pp. 35-41, Jul-Aug 1995.

[16] GE sol, [Online], available at: https://www.cbyge.com/products/sol

[17] Philips, [Online], available at: https://www.usa.philips.com/c-p/HF3550_-60/discontinued-wake-up-light/overview

[18] Ehren R. Dodson and Phyllis C Zee, "Therapeutics for Circadian Rhythm Sleep Disorders", *Sleep Med Clin*, **vol. 5(4)**, pp. 701-715, Dec 2010.

[19] Marshall Brain,"How Microcontrollers Work",*HowStuffWorks.com*[Online] ,available at: http://electronics.howstuffworks.com/microcontroller1.htm, 1 Apr 2000.

[20] "Arduino Due Specifications", *Arduino*, [Online], available at: https://store.arduino.cc/arduino-due

[21] "Intel Edison Specifications", *Intel*, [Online], available at: https://cdn-shop.adafruit.com/datasheets/EdisonDatasheet.pdf

[22] "Raspberry Pi Zero Specifications", *Raspberry Pi*, [Online], available at: https://www.raspberrypi.org/products/raspberry-pi-zero-w/

[23] "STM32F407VGT6 Specifications", *STM microelectronics*, [Online], available at: http://www.st.com/en/microcontrollers/stm32f407-417.html?querycriteria=productId=LN11

[24] Tom Tyson, "Serial EEPROM Solutions vs. Parallel Solutions", *Memory Products Division, Microchip*, [Online], available at: http://ecee.colorado.edu/ mcclurel/microchipan551.pdf

[25] Steve Drehobl, "Basic Serial EEPROM Operation", *Memory Products Division, Microchip*, [Online], available at: http://ecee.colorado.edu/ mcclurel/man536.pdf

[26] Jean-Michel DagaCaroline, PapaixMarylene CombeEmmanuel, RacapeVincent Sialelli, "Embedded EEPROM Speed Optimization Using System Power Supply Resources", *Lecture Notes in Computer Science*, **vol. 3254**.

[27] K. V. S. S. S. S. Sairam and N. Gunasekaran and S. R. Redd, "Bluetooth in wireless communication", *IEEE Communications Magazine*, **vol. 40(6)**, pp. 90-96, Jun 2002.

[28] "Resistive Touch Screen", *Baanto*,[Online], available at: http://baanto.com/resistive-touch-screen-technology

[29] "Capacitive Touch Screen", *Baanto*,[Online], available at: http://baanto.com/capacitive-touch-screen

[30] "Serial Peripheral Interface (SPI) ", *Sparkfun*,[Online], available at: https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi

[31] " Serial Communication ", *Sparkfun*,[Online], available at: https://learn.sparkfun.com/tutorials/serial-communication

[32] "I2C", *Sparkfun*,[Online], available at: https://learn.sparkfun.com/tutorials/i2c

[33] "Generic Standard on Printed Board Design", *ASSOCIATION CONNECTING ELECTRONICS INDUSTRIES*,[Online], available at: http://www.sphere.bc.ca/class/downloads/ipc_2221a-pcb%20standards.pdf

[34] Advanced Circuits, "Trace Width Website Calculator", [Online] available at: http://www.4pcb.com/trace-width-calculator.html

[35] Stephen G. Kochan, "Programming in C : A complete intorduction to the C programming language, Third Edition".

[36] Herbert Schildt, "Java: The complete reference, Seventh Edition".

[37] Energy Saver, "Type of lighting", *U.S Department of Energy*, [Onile] available at: https://energy.gov/energysaver/types-lighting.

[38] Warren J. Smith, "Modern Optical Engineering: The Design of Optical Systems"

[39] Atollic TrueSTUDIO, "Features", *Atollic TrueSTUDIO*, [Onile] available at: https://atollic.com/truestudio/features/

[40] Nextion, "Features", *Nextion*, [Onile] available at: https://nextion.itead.cc/

[41] Android, "Everything you need to build on Android", *Android*, [Onile] available at: https://developer.android.com/studio/features.html

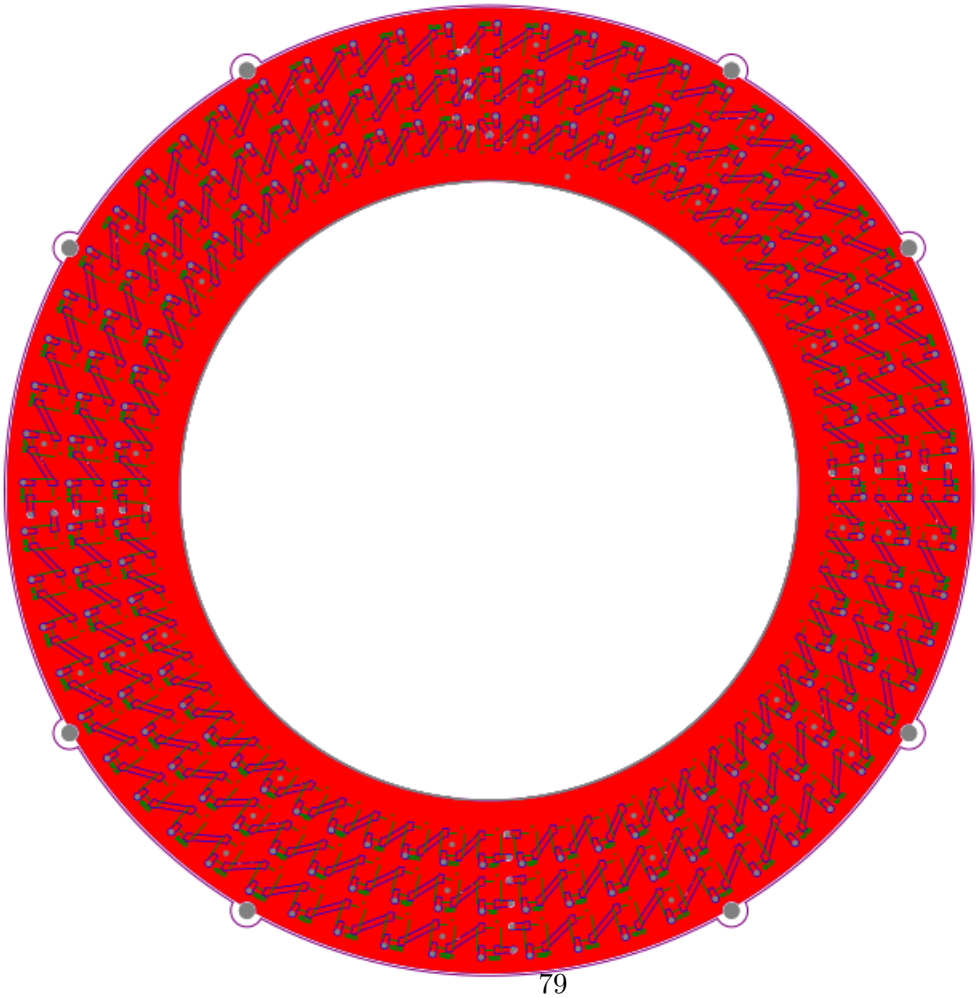
[42] authors, "Paper", *Journal*, **vol.**, pp.

Appendix A

Additional Files and Schematics

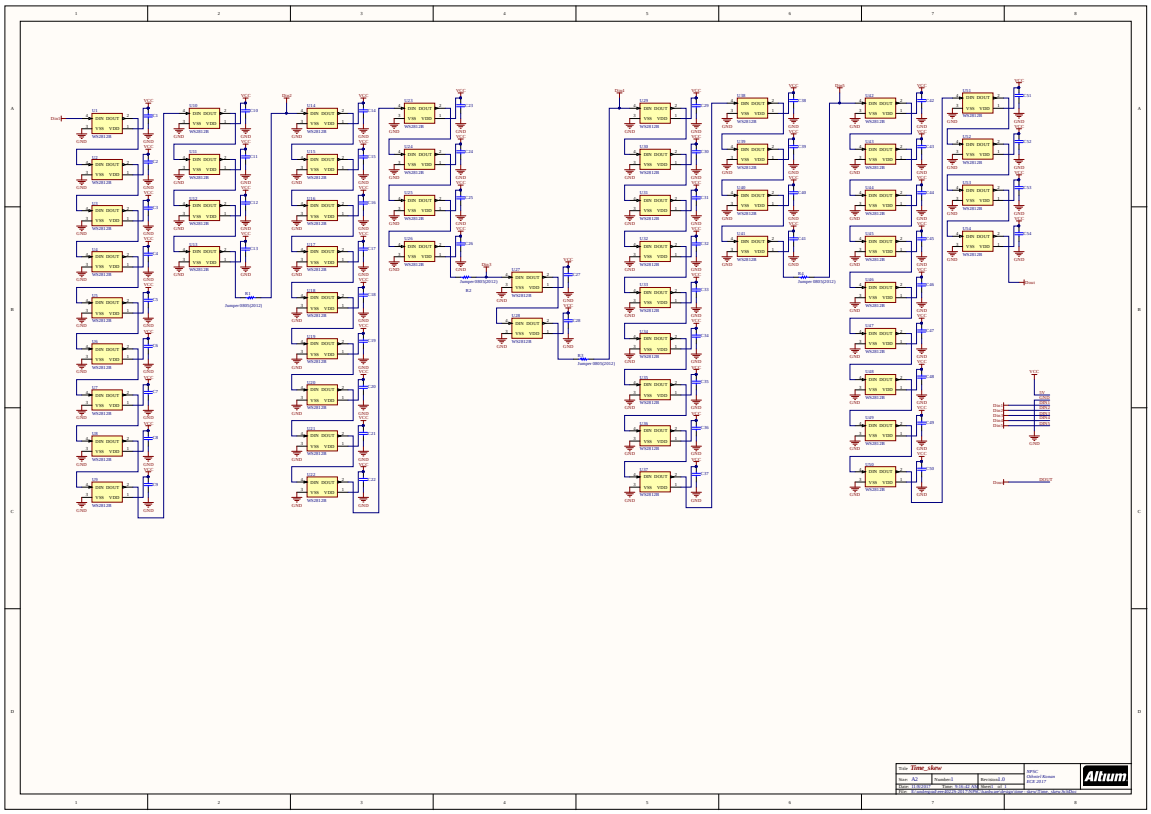
Add any information here that you would like to have in your project but is not necessary in the main text. Remember to refer to it in the main text. Separate your appendices based on what they are for example. Equation derivations in Appendix A and code in Appendix B etc.

(a)

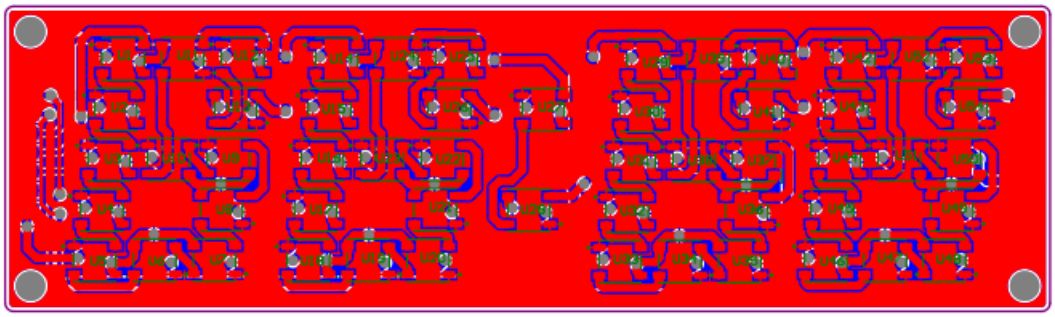


(b)

Figure A.1: Ring

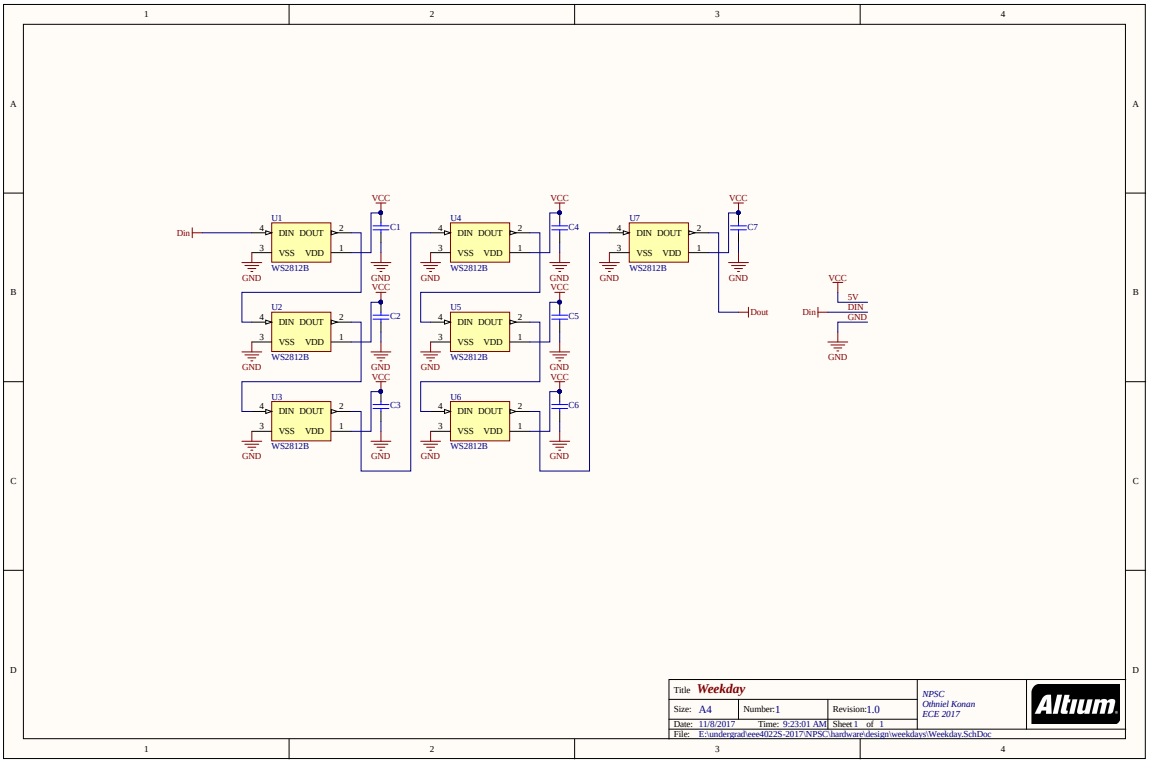


(a)



(b)

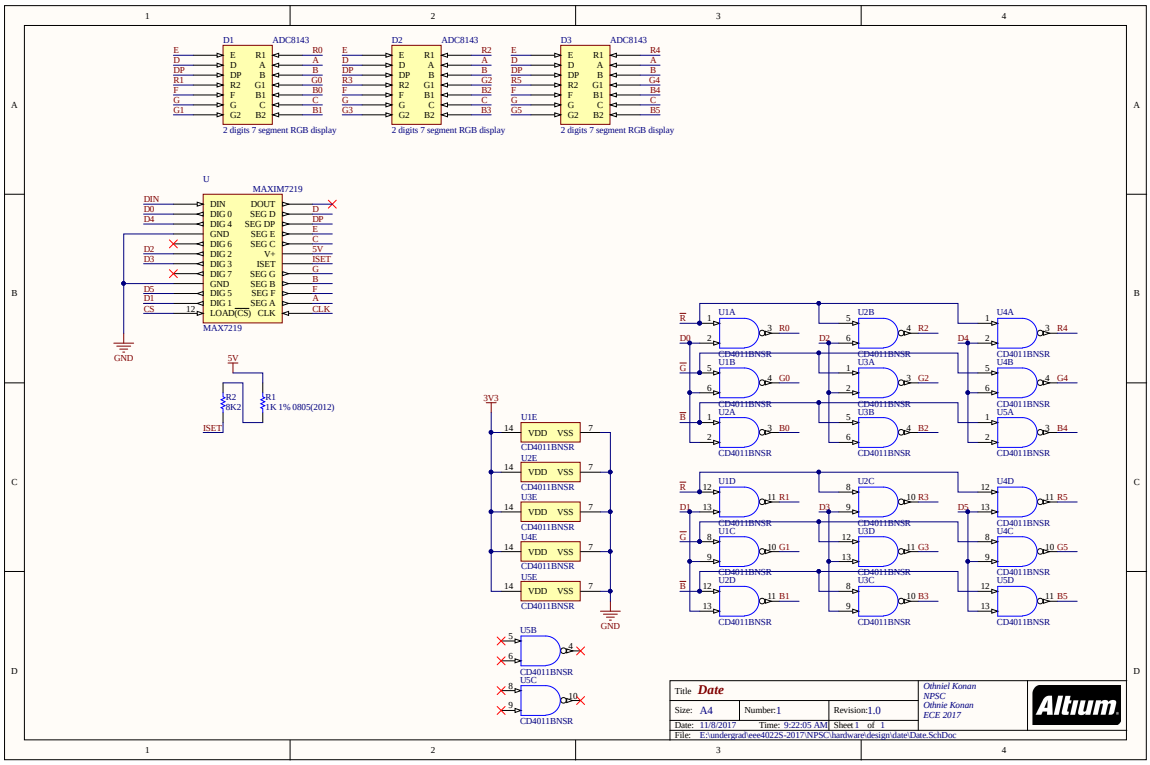
Figure A.2: Time



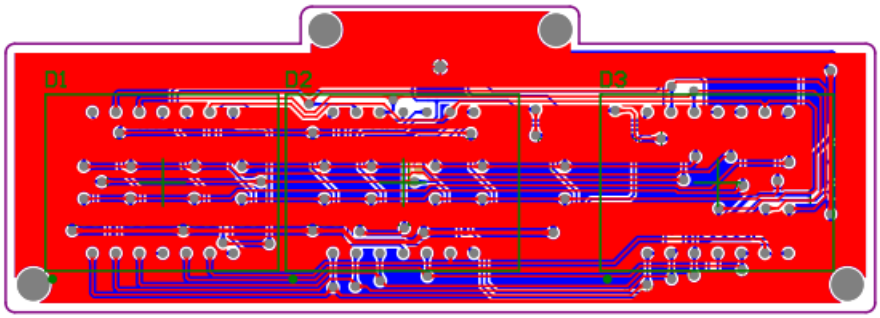
(a)

(b)

Figure A.3: Weekday



(a)



(b)

Figure A.4: Date

Appendix B

Datasheets

Appendix C

Addenda

C.1 Ethics Forms