ENEL200 Final Project

Sun Smart Project

Group 24: Neil Alombro, Stella Stiven, Bethany Kaye-Blake

Student Numbers: 53559923, 28578727, 59887386

Summary

The following report details the process of taking the problem of the high levels of skin cancer in New Zealand and designing a solution. This included defining a project goal, requirements, and specifications, then designing a prototype and testing it appropriately. The aim was to create a portable design that uses solar energy to power a UV detection device to increase education and awareness of UV radiation. The device displays the UV index through an exact integer display and a corresponding LED colour with the index ranges. The device then uses the UV index to estimate when users should apply sunblock and alert them by having a transducer beep and the LED flash red. The design uses a solar panel, Arduino UNO, and a UV detector as its main components. Overall, the prototype successfully achieves the original specifications and requirements. However, there is significant room for future developments in hardware, software, and purpose within the design.

Project Synopsis

Project: Solar powered sun smart alert device

Primary goal:

Create a device to alert users when they should re-apply sunscreen

Other key goals:

Increase New Zealanders awareness and education of UV radiation
Compact design which is portable
Utilise the energy from the sun when outside

Biggest uncertainty:

The consistency of power from solar panel

Other major uncertainties:

Sensitivity to outdoor elements, including wind, sand, and water
Whether the price of the solar panel is worthwhile

Most valuable requirements/features:

Solar panel able to provide sufficient power to device
UV detector for interpreting radiation
LED and 7-segment display representing of UV Index
Alert for users dependant on UV exposure
Optional:

Tradeoff priorities:

Low current draw from panel meaning simplified design
Arduino UNO expensive but keeps option open when designing and easier to implement
Sustainable power from solar panel vs most consistent power

Key deliverables:

Solar panel appropriate to design
Software created to facilitate design
Shield prototype for Arduino UNO
Mechanical stand holding solar panel and Arduino with shield cover

Key dates:

September 15th (Part orders)
Wednesday October 12th (Design review)
Friday October 21st (Group report and individual reflections)

Project Team:

Stella Stiven
Bethany Kaye-Blake
Neil Alombro

Expected user(s):

People spending time outdoors, including for summer sports, beach goers and sunbathers

Project value to user(s):

Alerts about sun protection when distracted
More awareness of UV radiation
Education around solar energy

Other important stakeholders/impacts:

Educational institutions who can use the device

Block diagram:

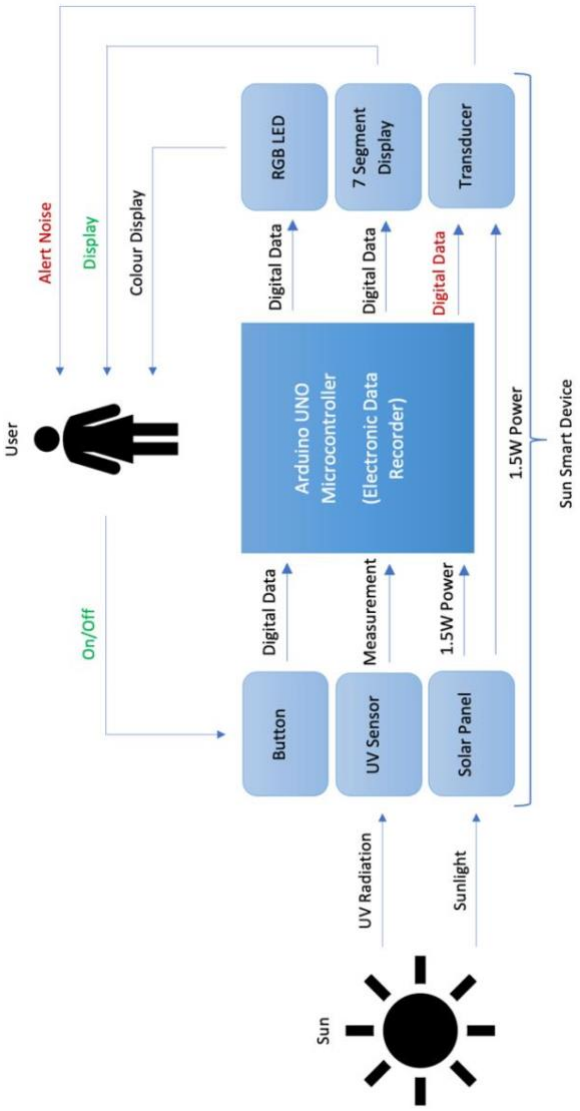


Table of Contents

1 Introduction

In New Zealand, almost 100,000 people are diagnosed with some form of skin cancer each year [1] making up 80% of all cancer diagnoses [2]. Resultant healthcare costs currently amount to almost $200 million each year, which is expected to grow to $295 million by 2025 [1]. With UV radiation 40% higher than similar latitudes in the Northern Hemisphere [3], our kiwi nature of 'she'll be right' does not help our increased need for sun smarts.

With these statistics, it is clear we need to be more proactive to protect our skin. This motivated our project design. Our project aimed to create a portable device that was capable of using the UV index to establish when a user should reapply sunscreen and alert them of this. Our users are essentially anyone outside in the sun including beachgoers, summer sports groups and sunbathers. Our current design consists of a collection of components on an Arduino prototype shield, an Arduino UNO microcontroller and a CAD stand. The first half of this report looks at the current problem and the goals, requirements, and specifications. The second half looks into the respective hardware, software, and CAD designs and how successful these are. A complete schematic for the electrical components, the CAD design and the Arduino code appear in Appendix A, B and C respectively.

2 Background

The design problem explored for our Final Project involved developing an educational tool to inform users about UV index levels and alert them when to seek more protection. The solution developed is built around a solar-powered Arduino which collects data from a UV sensor. This design uses the UV sensor to measure the wavelength of UV light and using a program, convert this measurement to the appropriate UV index. The UV index is used to determine the time between alerts in the form of a transducer buzzing. The design also includes a 7-Segment display and RGB LED. These two components display the current UV index as a number and colour respectively.

The Sun Smart Project was inspired by Kiwi summers and the rates of skin cancer in New Zealand. We decided to design a solution that would help educate and prevent people from being exposed to damaging levels of UV radiation. When researching how UV sensors work and can be used to educate, we were directed to Dr Martin Allen, a professor at the University of Canterbury. Dr Martin Allen specialises in UVB photodetectors for monitoring personal UV exposures for the prevention of skin cancer and vitamin D deficiency [4]. He has helped design an Electronic UV Dosimeter (Figure 1) which detects and measures UVB radiation exposure [5].

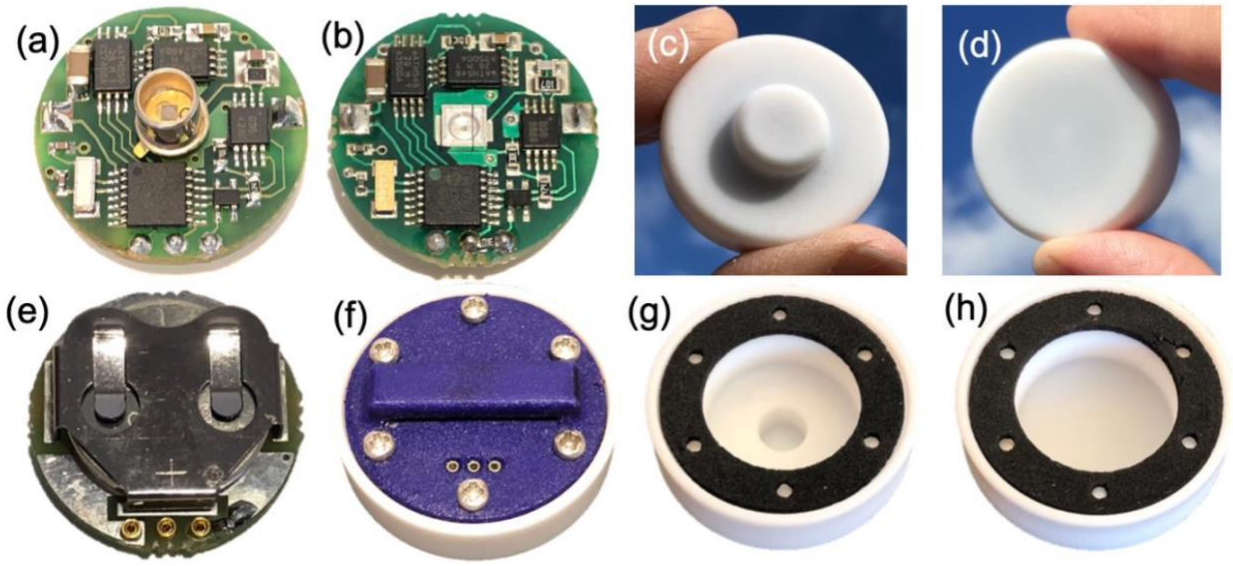


Figure 1: *EUV dosimeter construction* [5].

Although all UV radiation is harmful to humans, specific wavelengths are more harmful than others. UV sensors can be used to measure the wavelength of UV radiation. Photodiode-type UV sensors measure illuminance. When light strikes the photodiode, it energizes the electrons and causes an electric current. This electric current will be stronger in response to brighter light [6]. A wavelength-selective filter limits light striking the diode to only the UV region you are interested in detecting [7]. The two most damaging types of UV radiation are UVA and UVB. UVA radiation has a longer wavelength that is associated with skin ageing. On the other hand, UVB radiation has a shorter wavelength that is associated with skin burning [8].

3 Project Requirements

The overarching goal of this project was to increase New Zealanders' awareness and education about sun skin care. From this, we developed 3 key requirements as outlined below.

A. To use the UV index to track how much radiation the user has been exposed to.

B. To alert users when outside to reapply sunblock and/or find shade.

C. To utilize the energy from the sun when outside.

More specific requirements for individual aspects of the design are further outlined below.

Hardware Requirements

1. Sufficient power is supplied by the Solar Panel
    a. To the Arduino with
        i. the 7-segment display on
        ii. the LED on
        iii. the transducer on
        iv. and with all components on
    b. And a separate connection to the transducer
2. Has a visual indication of the UV index band
    a. LED has sufficient current to shine bright enough
3. Has a visual indication of the UV index number
    a. 7-Segment has sufficient current to display bright enough
4. Has an audible indication that the user has exceeded the UV index count
    a. The transducer is able to draw sufficient current to be heard
    b. The user is able to turn off the transducer
5. All components that users need to see or interact with can be seen and accessed
6. Shield connects to Arduino UNO
7. Components connect to appropriate Arduino pins
8. The solar panel is detachable from the rest of the hardware

Software Requirements

9. Modular structure to be applied to software
    a. Include individual component test program
    b. Include overall program design
10. Visually indicate UV Index
    a. Exact numerical UV Index
    b. Colour range corresponding to UV Index
11. Remind the user when to apply sunblock
    a. Easily detectable indicator
    b. Reminder calculated through research
    c. Ability to turn the indicator off once initiated

Stand Requirements

12. Holds components securely

13. Holds solar panel at an optimal angle to get energy from the sun
14. Keeps components from shading/obscuring the solar panel from the sun

Constraints

As well as the above requirements, we also have a series of constraints outlined below:

15. The number of Arduino pins
    a. 14 digital input/output pins (6 of these pins can be used as PWM outputs)
16. Space on the Arduino prototype shield to solder components
17. SCL and SDA Arduino pins need to be used for the UV sensor
18. Delivery lead time for components
19. UV Sensor not obscured
20. The typical supply of solar panels is 5.5V and 270mA

4 Hardware Specifications

Using the above hardware requirements, the following specifications for the design have been developed.

1. The Solar panel must provide
    a. More than 5V to the Arduino
    b. At least 200mA to Arduino as the UNO and I/O pins can draw up to 200mA
    c. An external connection to the VDD of the transducer as the Arduino cannot supply sufficient current (requires >20mA which is the standard I/O pin current)
2. The Solar panel is connected to the prototype shield using male-to-female connector pins so it can be disconnected
3. RGB LED provides a visual indication of the UV index band
    a. Connected to 3 PWM pins of the Arduino to encode the different colours
    b. Current-limiting 220Ω pull up resistors are connected to the three RBG LED input pins
    c. Not obscured by other components so the index can be seen
4. 7-segment display visually communicates the current UV index number
    a. 0-9 indicate index level and H indicates an index ≥ 10
    b. Connected to 7 Arduino standard I/O pins
    c. GND connected pins of the display through a 330Ω resistor
    d. Not obscured by other components so the index can be seen
5. Transistor is able to alert the user
    a. Is connected to an Arduino PWM pin to create changing differences over buzzer pins
    b. Has a 1KΩ resistor across buzzer pins
6. Button
    a. Is connected to an Arduino input pin
    b. Is not obscured by other components so can be pressed
7. UV detector is connected to Arduino SDL and SVA pins for communicating
8. Prototype shield pins can be connected to Arduino UNO pins

5 Software Specifications

9. Set up global variables and initialise accordingly for the following
    a. Transducer pin number
    b. RGB LED pin numbers
    c. 7-segment display pin numbers
    d. Button pin number
    e. Time
        i. Overall time
        ii. Number of minutes elapsed
        iii. Number of minutes since last sun protection indicator
        iv. Time threshold for how often the reminder should occur.
    f. UV data
        i. Raw measurement
        ii. Calculated index
    g. Transducer Boolean value for producing sound.
10. Components are initialised
    a. UV Sensor initialised with the following functions
        i. Set an instance of Adafruit_LTR390 as ltr with the function Adafruit_LTR390()
        ii. ltr.begin()
        iii. ltr.setMode() with mode set to UV sensor
        iv. ltr.setGain() with sensor gain set to 3
        v. ltr.setResolution() with resolution set to 16-bit
        vi. ltr.setThresholds() with thresholds set from 100 to 1000
        vii. ltr.configInterrupt() with interrupt enabled and set to UV sensor mode
    b. Transducer set as output.
    c. RGB LED set as outputs.
    d. 7-segment display set as outputs.
    e. Button set as input.
11. Create helper functions to simplify programs
    a. RGB LED function to analogWrite() new values to the corresponding pins.
    b. Display functions to display numbers zero to nine, letter H, and clear display.
    c. Display to 7-segment and RGB LED to switch displays dependent UV index and range.
12. Set up an ongoing loop
    a. Update the overall time
    b. Check if a minute has elapsed and, if so
        i. Update raw UV measurement and UV index
        ii. Display new RGB LED colour range
        iii. Add UV index points to the accumulator
    c. Check if UV index points have elapsed 100 points and, if so
        i. Set transducer to produce noise
        ii. Flash red on RGB LED
    d. Read button state

i. If the button is pressed and the transducer is making noise, turn off and reset the UV index points
ii. If the button is pressed and the transducer is not making noise, have the 7-segment display the current UV index
iii. If the button is not pressed, clear the 7-segment display.

6 Stand Specifications

13. All components must fit tightly into their holders and not fall out of the holders when the stand is inverted
14. Solar panel holder must hold the solar panel at 23 degrees for optimal angle to get energy from the sun, this ensures requirement 13 is met
15. The solar panel holder and Arduino holder clip together and cannot be pulled apart
16. The hinge between holders has no resistance when rotating the Arduino holder about the hinge

7 Design Overview

Our overall design has four main components. The power supply from the solar panel, the hardware, the Arduino and software, and the printed stand.

The design itself, using power from the panel, uses the UV sensor to read the wavelength of UV radiation. This reading is communicated to the Arduino once every 60 seconds where it is interpreted into an index reading through calculation. The RGB LED and 7-segment display will correspond to the reading. The transducer will remind the user to apply sunscreen once a certain amount of time has elapsed (using the UV index in the calculation).

Hardware

In terms of hardware, there are four parts the electrical components fall into. The first is power, which, in our design, comes from the 1.5W solar panel [9] with a typical voltage and current of 5.5V and 270mA respectively. The solar panel is connected to the Arduino barrel jack, as well as the positive lead connected to the VIN of the transducer.

The second part is the inputs that send information to the Arduino UNO. The two components in this section are a simple tactile button [10], which allows the user to interact with the device, and the LTR390 UV sensor [11]. The UV sensor measures the UV radiation wavelength and, using the SDL and SVA pins of the Arduino, sends this to be interpreted by the software.

The third section of the hardware design is the alert. This alert is to signify to the user that they need to do something to protect themselves from UV, whether it be applying sunscreen or finding shade. The electrical component used is the piezo buzzer transducer [12] that, using an Arduino PWM pin, buzzes until switched off with the button.

The final part of the design consists of the components that communicate information to the user. The 7-segment common cathode LED display [13] can be turned on to display the current UV index that is read by the sensor and interpreted by the Arduino. The RGB LED [14] is always on when the device is in use and represents the UV index using the correlating colour. The colour is also determined by the UV sensor reading and set by the Arduino. Figure 2 is a block diagram depicting the hardware relationships laid out above.

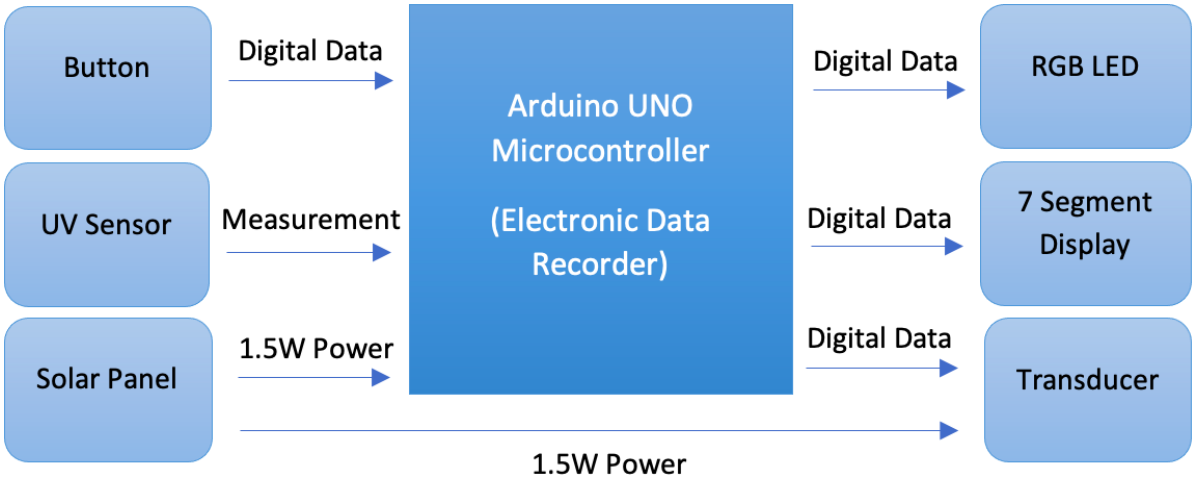


Figure 2: *Block diagram of hardware design*

Software

In terms of software, the design includes a modular system. Included are individual component tests with the hardware and an overall program to fulfil the requirements and specifications outlined above. The individual component programs helped build up the overall program using cascading for the related inputs and outputs. There are three main sections of the overall program's Arduino code. This includes initialisation, helper functions, and the main loop. The entirety of the program is included in Appendix C.

Within the initialisation, this required setting up global variables for the components' associated pin numbers for the program to use. The initialisation also required global variables storing time variables and the UV sensor's data and calculated index. The setup included the mandatory functions in the 'Adafruit_LTR390.h' header library file. This included setting the mode to UVS (UV sensor). The specific pins were also assigned if they were an input or an output to allow for digital and/or analog reads or writes later in the program.

The helper functions for the software include an 'RGB_color' function that adjusts the red, blue, and green light ratios to correspond to the current UV index range. The 7-segment display also requires many helper functions with the different pin configurations each specific display requires. Controlling the RGB LED and the 7-segment display also required a helper function consisting of a switch case.

The main section of the software was the loop. This conducted the fulfilment of most of our specifications. This loop was constantly updating. We began with checking the time and updating the variable 'overall' with the millis() function. An if statement then checks if a minute has elapsed from previously and updates the 'uv_data' and 'uv_index' variables accordingly. The helper function is also called to update the RGB LED colour. The UV index points system is then checked if it has elapsed the 100 points. If so, this buzzes the transducer to remind the user to protect themselves from the sun and the RGB LED flashes red. The button is also checked if it has been pressed. If pressed when the transducer is producing noise, then the button acts to stop this noise. If pressed when the transducer is not producing noise, then the button will turn on the 7-segment display.

The UV index points system is a key design aspect of our device. Through research [15], we have determined, with a safety margin, that if we treat a UV index to be done in points per minute accumulation that the recommendation averages to be 100 points for the user to need sun protection. This also means that we take in an average of UV indices through a period of time as the UV index updates every minute and accumulates accordingly. The state machine that describes this is as follows.

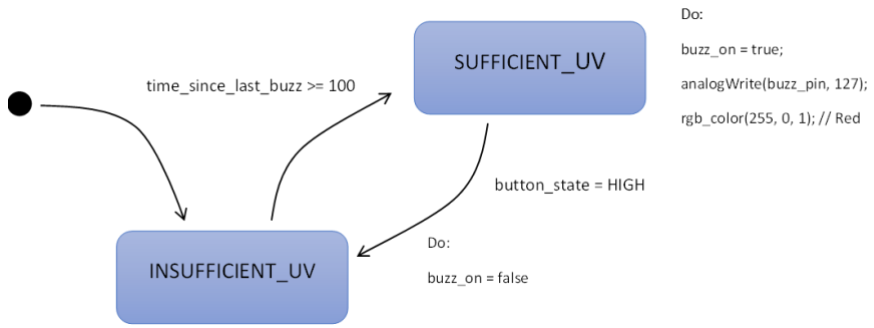


Figure 3: *Simple state machine for software showing device behaviour with sufficient/insufficient UV.*

8 Design Rationale

A solar panel was used to supply power to the Arduino and shield. The reason a solar panel was used as a power source was to utilise the energy from the sun. Since our design is intended to be used outside, and on reasonably sunny days when the user would be concerned about sun damage, our rationale was to utilise the sun's energy to power our design solution. The solar panel used was a Monocrystalline Solar Cell with an average voltage and current rating of 5.5V and 270mA respectively [9]. The reason this solar panel was used was that it reached the minimum voltage rating for the Arduino 5V, and the maximum current draw the Arduino would require of approximately 200mA. The solar panel was also chosen because it could provide enough external voltage sources to the transducer, which requires more than the standard I/O pin current of 20mA.

The Arduino Uno was used as the microcontroller for our design because our group had experience working with this microcontroller. The Arduino was also compatible with the Arduino porotype shields supplied in the UC electronics store which we had easy access to for designing and building our shield. The Arduino was also compatible with the UV sensor our design was going to use. The last reason the Arduino was used is that it has PWM pins and hence could change the difference over buzzer pins which was required for the transducer to produce sound.

An LTR390 Ultraviolet Sensor was used to measure the wavelength of UV light and then our program was written to convert these measurements into a UV index. This sensor was used because it was compatible with an Arduino. The pins required to operate the sensor were available on the Arduino and the Arduino code included a function to convert the wavelength to the appropriate UV index. Another reason this UV sensor was used is that it was relatively inexpensive at $9.62 and detected UVA radiation which causes skin ageing. We later discovered that UVB radiation is more harmful to humans as it causes burning but accurate UVB sensors are harder to source and more expensive ($87.41 [16]) so would not have been an option for this project with our budget of $50. In the future, it would be easy to swap out the sensor we used for a more accurate UVB sensor which is more appropriate for our design. For the display, our design used a 7-segment display and an RGB LED (Figure 4).

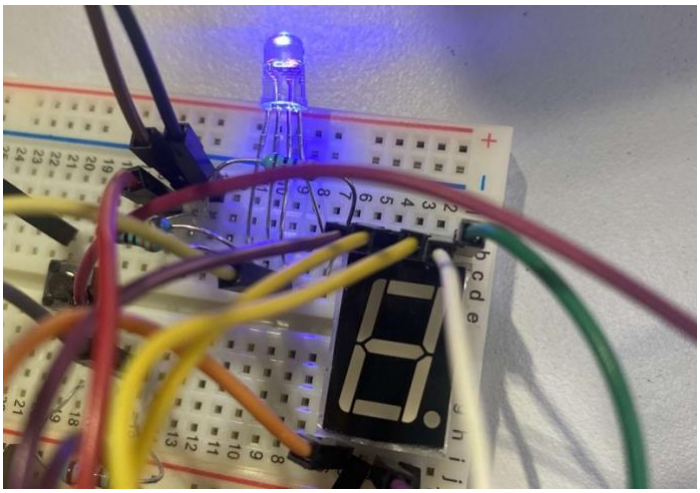


Figure 4: *7-Segment display and RGB LED circuit when the UV index is 10+ and the button is not pressed.*

The 7-segment display was used to display the UV index number when a button was pressed. The decision for the display to only be on when the button was pressed was to reduce power consumption. The rationale for using one 7-segment display over a more complex display, such as an LCD, was because of the constraint of the number and type of pins on the Arduino available to use. The UV sensor used the SDL and SVA pins so only the PWM, and standard I/O pins were available. One 7-segment display used 7 pins, so having multiple would have required more than on the Arduino itself. The RGB LED was used to communicate the UV index via colour. This decision was made because we wanted a constant display of the UV index for the user to quickly be able to see from a distance. The LED also flashed when you need to seek more protection from the sun. This design decision was made so if the user was unable to hear the alert, they would notice the LED alerting them instead.

A piezo buzzer transducer was used for the alert (Figure 5). The transducer was used because it was available at the UC electronics store. The transducer was chosen because it could make a loud sound which is required to make sure the user hears it and knows to seek more protection from the sun.

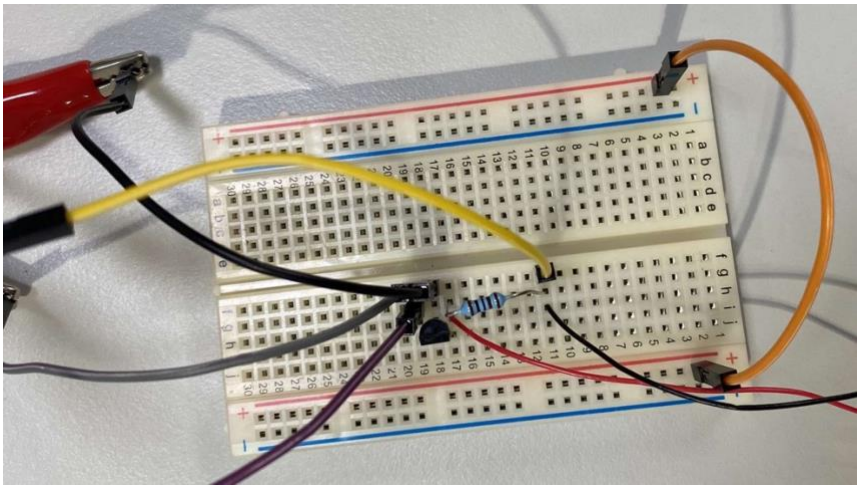


Figure 5: *Piezo buzzer transducer circuit.*

After our demonstration, the idea of supplying the user with a silent mode was mentioned. After consideration, we decided this would likely not be necessary for a couple of reasons. Firstly, the point of our design is to alert the user when they need to reapply sunblock or seek shade from the sun depending on the UV index and the time at which they have been exposed. Having a silent mode would defeat the purpose of this. Secondly, our design is intended to be used outside, and therefore in places which do not require silence such as at the beach, pools, or sports events. Having an audible alert in these environments should not be an issue.

The stand was designed (Appendix B) so that the Arduino/shield system could be folded up under the solar panel holder. This design decision was made to make the system more compact and portable. Because the intention of the design is to be used on sunny days when the user is spending time outside, such as going to the beach, the design needs to be portable. The rationale behind the 23-degree angle for the solar panel holder is that this is within the optimal angle range for a solar panel to absorb energy from the sun [17]. The lower end of the angle range was chosen so that the solar panel was not too high and cast shadows onto other components, like the UV sensor, which also need access to direct sunlight. The solar panel was also positioned higher than the Arduino, so it had an unobstructed view of the sun. The UV sensor was also positioned on the furthest end of the shield away from the solar panel for the same reason. This ensured the UV sensor was in the best location on the shield to get unobstructed access to direct sunlight.

Images of the final design with all components attached can be found in Appendix D. An annotated model of the stand design can be found in Appendix B.

The software was designed with modularity in mind to make the design of the overall program easier and alterable for future development. The main design concept within our software was the ability to remind the user to protect themselves from the sun. This indication was computed through the accumulation of a ‘UV Index Points System’ that was explained in the software section of the design overview. The tradeoff for implementing this was the lack of research that went into this system. It was easy to implement with the minimalism of a never-changing threshold but even the threshold implemented has many caveats. The system’s simplicity with the displays chosen also meant that there was no conduit to be able to consider the user’s skin type, the type of sun protection (sunblock, sun spray, seeking shade, etc.), the SPF rating (if sunblock was used), and much more.

9 Testing

The following tables detail the testing that was done for each part of the design. Each test gives a short description of it and the inputs. It explains the expected outputs and then the actual outputs from the test. Then the related specifications and requirements are noted in the final column.

Stand

The tests on the stand are as follows in Table 1.

Table 1: *Test cases for the stand.*

Test	**Inputs**	**Expected Outputs**	**Actual Outputs**	**Specification/ requirement**
Solar panel slides into holder	Slide solar panel through the top of holder	Solar panel is tightly fitted	Solar panel is tightly fitted	Requirement 12
Arduino slides into holder with shield attached	Attached shield to Arduino and slide Arduino into holder	Arduino is tightly fitted	Arduino is tightly fitted	Requirement 12
Measure angle that solar panel is being held up to sun	Measure angle of solar panel holder on SolidWorks	Solar panel holder has a 23-degree angle from the ground	Solar panel holder has a 23 -degree angle from the ground	Specification 14
Components do not slide out of holders when stand is inverted	Invert stand	Components do not fall out of holders	Components do not fall out of holders	Specification 13
The Arduino holder and solar panel holder clip together tightly	Clip Arduino holder into solar panel holder clips and try to pull apart	Holders clip together without breaking and do not separate when pulled in separate directions	Holders clip together without breaking and do not separate when pulled in separate directions	Specification 15
Hinge allows the holders to move freely	Rotate holders about hinge	Holders rotate smoothly about hinge	Holders rotate smoothly about hinge	Specification 16

Hardware

The tests on the hardware are as follows in Table 2.

Table 2: *Test cases for the hardware.*

Test	**Inputs**	**Expected Outputs**	**Actual Outputs**	**Specification/ requirement**
Solar panel placed outside with a multimeter across +ve and gnd leads	Mildly sunny day	Multimeter should read at least 5.5V	Multimeter read 5.8V	Specification 1.a
Solar panel connected to prototype shield	Panel male to female connector pins removed	Leads can be disconnected and panel detached from shield	Leads can be disconnected and panel detached from shield	Requirement 8 Specification 2
Prototype shield is placed on Arduino	Pins and sockets are aligned and pushed together	Pins and sockets fit together and all required pins can be used as normal	Pins and sockets fit together and all required pins can be used as normal	Requirement 6 Specification 8
Button is connected as in the schematic with LED test code uploaded to the Arduino	Button is pushed	LED changes from red to green	LED changes from red to green	Requirement 4.b, 5 Specification 6.a
Transducer VDD connected solar panel and to MOSFET with Arduino PWM pin	Panel outside with >5.5V, Arduino outputs signal with PWM at 50%	Transducer buzzes periodically	Transducer buzzes periodically	Requirement 1.b, 4.a Specification 1.c, 5.a
UV Detector is connected up to SDL and SVA pins	UV Detector is placed outside and UV test code is uploaded to the Arduino	Arduino receives the readings and prints them on a constantly updating screen	Arduino receives the readings and prints them on a constantly updating screen	Requirement 7 Specification 7
7-Seg display can show digits 0-9 and H	Arduino 7-seg test code is run	Display can be seen and numbers are read from 0 through to 9, then ending on H	Display can be seen and numbers are read from 0 through to 9, then ending on H	Requirements 3.a, 5 Specification 4

RGB LED can indicate index colours when connected to 3 PWM pins	Arduino RGB LED test code run that manipulates colour using PWM pins	LED should run through index colours from green, to yellow, to orange, to red & to purple and can be seen by observer	LED should run through index colours from green, to yellow, to orange, to red & to purple and can be seen by observer	Requirements 2.a, 5 Specifications 3.a 3.b

Software

The tests on the software are as follows in Table 3.

Table 3: *Test cases for the software.*

Test	**Inputs**	**Expected Outputs**	**Actual Outputs**	**Specification/ requirement**
Conduct transducer component test	Loop analogWrite() 127 and 0 into transducer pin with delay	Transducer buzzes intermittently.	Expected output and delay was accurate at one second in-between.	Requirements 9.a, 11.a Specifications 9.a, 9.g, 10.b
Conduct RGB LED component test	Loop through different colors with helper function to change LED light output with delay. Example: rgb_color(0, 255, 0) outputs green	Changes colour from green, yellow, orange, red, and purple with delay.	LED Colour changed through the indicated colour ratios.	Requirements 9.a, 10.b Specifications 9.b, 10.c, 11.a
Conduct 7-segment display component test	Loop through different displays through helper functions that go from numbers zero to nine, and the letter H. Example: display_zero()	Loops through displaying numbers zero to nine, the letter H, and keeps repeating.	Looped through the necessary displays.	Requirement 9.a, 10.a Specifications 9.c, 10.d, 11.b
Conduct button component test	Loop through checking button state. When pressed, display will loop through index numbers and the	When button is pressed, the index looped and the number display and colour matched the universal UV index indicators.	Number display and colour indicator matched perfectly and in sync.	Requirement 9.a Specifications 9.d, 10.e

	LED will correspond to the given range.	Example: When display was 0, 1, and 2 the LED was green.		
Conduct UV Sensor component test	Test through printing on serial monitor the live raw measurements and calculated UV index.	UV index on serial monitor, when outside, corresponds to observed UV index.	Matching UV indices on serial monitor and from live weather data online.	Requirement 9.a, 11.b Specifications 9.f, 10.a
Test overall program with manual UV indices and reduced time thresholds	Run program multiple times with different values set for 'uv_index' variable.	Display with number and colour to be correct to the various UV indices tested and noise produced accordingly. Example: uv_index = 7 TIME_THRESHOLD = 20 Display number 7 with RGB LED on orange. Noise produced and red flash every three minutes after button push.	Example at 'uv_index' set as 7 is met. Other indices also meet required indicator displays.	Requirement 9.b, 10.a, 10.b, 11.a, 11.c Specification 9.e, 11.c, 12.a, 12.b.ii, 12.b.iii, 12.d.i., 12.d.ii, 12.d.iii
Test overall program with UV sensor input and reduced time thresholds	Run program with input of raw measurements from the UV sensor and a reduced TIME_THRESHOLD of 20.	Dependent on observed UV index; noise and red flash produced accordingly. Also, button push without noise should show 7-segment index. RGB LED should always be on.	Tests ran when UV index observed was 5. Noise and a red flash were produced every four minutes after the button push. Button pushed without noise showed 7-segment display a number '5'. RGB LED showed yellow.	Requirement 9.b, 10.a, 10.b, 11.a, 11.b, 11.c Specification 12.b.i
Test overall program with UV sensor input and	Run program with input of raw measurements from the UV sensor and normal	Dependent on observed UV index; noise and red flash produced accordingly. Also,	Tests ran when UV index observed was 5. Noise and a red flash were produced every 20	Requirement 9.b, 10.a, 10.b, 11.a, 11.b, 11.c

normal time thresholds	TIME_THRESHOLD of 100. Note: This test will take longer than the others.	button push without noise should show 7-segment index. RGB LED should always be on.	minutes after the button push. Button pushed without noise showed 7-segment display a number '5'. RGB LED showed yellow.	Specification 12.c.i, 12.c.ii

10 Evaluation

The problem our design aims to solve is ignorance about sun damage. Our group addressed this problem by designing an educational tool to inform users about UV index levels and alert them when to seek more protection.

Referring to the testing section above, it can be determined whether the components making up our design work as expected and met their specifications and requirements. The main components tested and evaluated were:

- Stand
    - All requirements and specifications outlined previously were met by the stand designed. The final design presented in this report was the second version designed and 3D-printed. The first prototype had some measurement errors, and the clips were too thin causing them to snap when clipping the two parts together, as seen in Appendix B.
- Hardware
    - Specification 1.a, under hardware, was that the solar panel would supply more than 5V to the Arduino. This requirement is met on mildly sunny days, as seen in Table 2, but on days where the UV index sits around 1-2 this requirement struggles to be met. Although our design is intended to only be used where the user would be worried about skin damage from UV radiation, we wanted our design to still be used on days when the damage may be less significant. To solve this issue, our design can also support alternative power supply options which include a computer, being attached through the Arduino, and a 9V battery.
    - The sensor used in our design detects UVA radiation. Humans are sensitive to this type of radiation, as it causes skin ageing. However, after doing more research into UV radiation damage, it was discovered that the sensor we used did not detect UVB radiation which is associated with skin burning. For our project, it would have been more appropriate to use a sensor which detects UVB radiation. Although accurate UVB sensors are more expensive and harder to source, this type of sensor would have been more appropriate for our design. Realistically we could swap out the sensor we used for a UVB-specific sensor.

- Software
    - All requirements and specifications outlined previously were met by the software designed through component and overall programs. Design works perfectly with what was specified and tested through simulated UV indices, simulated smaller thresholds, and with our final hour-long test with real-time UV measurements and researched thresholds. All the components work in unison and cascade inputs and outputs to make a working device.
    - Current system for determining the need for sun protection is flawed with our minimal research and limited means with time for the project. Future developments should be made so we consider other factors that affect this time and if this can be proliferated for commercial use without legal and/or ethical problems.
    - Current evaluation sees software could be iterated in the future easily with more components being integrated and other interfaces. More upskilling of our team would be required if components that we had no experience in were added.

When evaluating our design, it meets all outlined goals, requirements, and specifications. Therefore, our design achieves its purpose. However, we have realised through this evaluation process that our design still lacks in ways that our requirements and specifications could not capture. The following outlines the improvements we would make in a future iteration of this product.

Design

Future improvements include:

- Product development costs
    - Microcontroller choices for a cost that is better utilised. Could dismantle and use relevant parts of Arduino microcontroller.
    - Looking for more cost-effective solar solutions.
- Product development hardware
    - Developing consistency of voltage from the panel and looking into batteries that the solar panel could charge while excess energy is being supplied from the sun.
    - Display screen having more information
        - Possible Bluetooth feature that utilises users' phones as a display screen.
        - Replacing the 7-segment display with an LCD.
- Product development features
    - Feature to include skin type so exposure time can be specific to the user.
    - Feature to enter the SPF of sunblock applied, this would affect the exposure time until the user needs to seek more sun protection.

- Make the design wearable
    - Attachments for various applications such as ski helmets, bikes, hats, and clothes for other outdoor physical exercises.
    - Would need to consider where the design is attached, as different areas are exposed to more/less sun e.g. if the device was attached to the wrist it would not be accurate at alerting the user when to seek sun protection if their shoulders were more exposed to the sun during the time. Could implement a safety margin for different parts of the body.

Ideas to consider in future designs:

- Data
    - Ambient vs UV light. Teach users that UV radiation still exists even when there is no ambient light. We could measure the different types of radiation and incorporate this into our display (or measurement, data, etc. if we go the Bluetooth route).
    - A short falling of using a sensor to detect UV radiation is that it does not consider that radiation hits our skin at different angles. When using a sensor, the angle of radiation it detects is limited to a small range on either side of 90-degree. This result in some error when trying to measure and simulate the damage to humans caused by UV radiation. Unfortunately, this is relatively unavoidable when using a sensor.
- Audience
    - Could target children to teach about sun safety and solar energy on solar cell output with the same device. This idea was recommended to us by Dr Martin Allen, he is developing a similar device to ours and mentioned the idea that renewable solar energy could be taught alongside UV exposure if the design would be powered by a solar panel. This would be a good way to educate the younger generation about renewable energy as energy generation is becoming a more relevant subject with population growth and climate change [18].
    - For legal reasons, if this device was sold commercially, it might be best not to have the device tell the user how many minutes they have left in the sun until needing to seek protection. This is because the calculation depends on a lot of variables such as skin type, sunblock SPF, the dosage of SPF and part of the body. This means the calculation is rarely accurate and the manufacturer could face problems when users get burnt using their product.

References

[1] Sun Smart, “Skin Cancer Facts and Figures,” The Cancer Society, 2022. [Online]. Available: https://www.sunsmart.org.nz/skin-cancer/facts-and-figures/. [Accessed 11 October 2022].

[2] Sun Smart, “Skin Cancer,” The Cancer Society, 2022. [Online]. Available: https://www.sunsmart.org.nz/skin-cancer/. [Accessed 12 October 2022].

[3] BPAC NZ, “Reducing the burden of melanoma in New Zealand,” BPAC, 17 January 2020. [Online]. Available: https://bpac.org.nz/2020/melanoma.aspx. [Accessed 10 October 2022].

[4] The University of Canterbury, “Martin Allen,” [Online]. Available: https://www.canterbury.ac.nz/engineering/contact-us/people/martin-allen.html. [Accessed 20 October 2020].

[5] N. S. K. M. N. B. L. a. R. L. M. Martin W. Allen, “Use of Electronic UV Dosimeters in Measuring Personal UV Exposures and Public Health Education,” *MDPI,* 2020.

[6] Davis Instruments, “What is a UV sensor?,” [Online]. Available: https://www.davisinstruments.com/pages/what-is-a-uv-sensor#:~:text=A%20photodiode%2Dtype%20UV%20sensor,a%20digital%20or%20analog%20output.. [Accessed 20 October 2020].

[7] Vernier, “UVB Sensor User Manual – Vernier,” [Online]. Available: https://www.vernier.com/manuals/uvb-bta/. [Accessed 20 October 2020].

[8] H. Alexander, “What’s the difference between UVA and UVB rays?,” MD Anderson Cancer Center, [Online]. Available: https://www.mdanderson.org/publications/focused-on-health/what-s-the-difference-between-uva-and-uvb-rays-.h15-1592991.html. [Accessed 20 October 2020].

[9] seeed studio, “1.5W Solar Panel 81X137,” seeed studio, 24 February 2019. [Online]. Available: https://media.digikey.com/pdf/Data%20Sheets/Seeed%20Technology/313070002_Web.pdf. [Accessed 17 October 2022].

[10] Mountain Switch, “Tactile Switches,” Mountain Switch, 27 May 2012. [Online]. Available: https://www.arduino.cc/documents/datasheets/Button.pdf. [Accessed 17 October 2022].

[11] A. Torres, “Adafruit LTR390 UV Sensor,” Adafruit Industries, 15 November 2021. [Online]. Available: https://cdn-learn.adafruit.com/downloads/pdf/adafruit-ltr390-uv-sensor.pdf. [Accessed 19 October 2022].

[12] Adafruit, “Large Enclosed Piezo Element w/Wires,” Adafruit, 21 January 2016. [Online]. Available: https://media.digikey.com/pdf/Data%20Sheets/Adafruit%20PDFs/1739_Web.pdf. [Accessed 17 October 2022].

[13] Octopart, “FND500 7-Segment Numeric LED Display,” Octopart, 2020. [Online]. Available: https://datasheet.octopart.com/FND500-Fairchild-Semiconductor-datasheet-41701481.pdf. [Accessed 19 October 2022].

[14] Inolux, “HV-5RGB25 RGB LED,” Inolux, 21 July 2015. [Online]. Available: https://www.inolux-corp.com/datasheet/Inolux%20Lamp/TH%20Lamp/HV-5RGBXX%205mm%20Full-Color%20Series.pdf. [Accessed 19 October 2022].

[15] “UV Index: The Sun Safety Scale,” Premier Dermatology, [Online]. Available: https://pdskin.com/blogs/uv-index-the-sun-safety-scale. [Accessed 1 October 2022].

[16] DigiKey, “MIKROE-4144,” [Online]. Available: https://www.digikey.co.nz/en/products/detail/mikroelektronika/MIKROE-4144/12547439?utm_adgroup=Evaluation%20Boards%20-%20Expansion%20Boards%2C%20Daughter%20Cards&utm_source=google&utm_medium=cpc&utm_campaign=Shopping_Product_Development%20Boards%2C%20Kits. [Accessed 20 October 2020].

[17] Zen Energy, “Zen Energy - How do you know if your roof is good for solar?,” [Online]. Available: https://www.zenenergy.co.nz/blog/how-do-you-know-if-your-roof-is-good-for-solar#:~:text=The%20best%20tilt%20angles%20for,is%20lower%20in%20the%20sky).. [Accessed 20 October 2020].

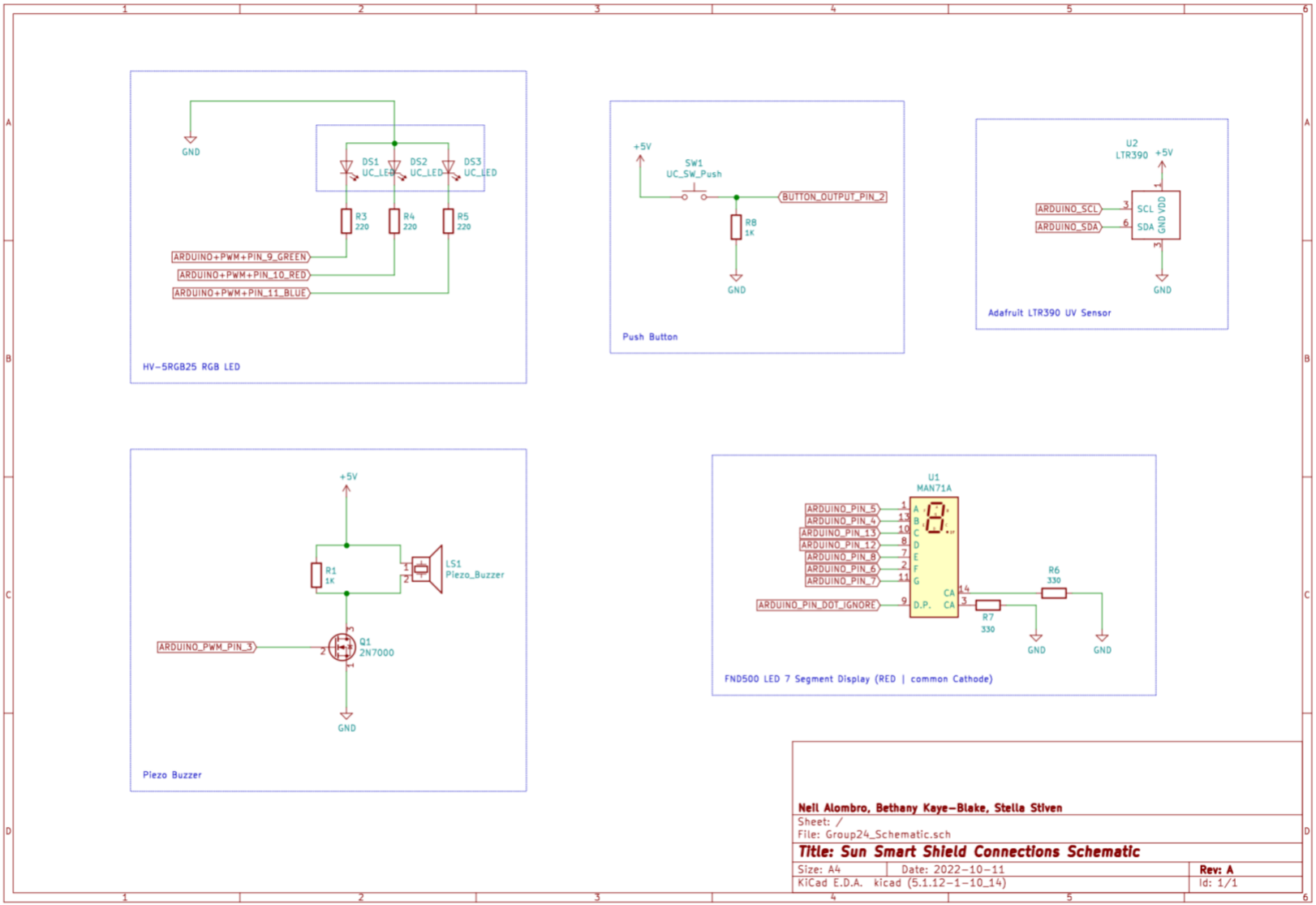
[18] World Wildlife Fund, “Importance of Renewable Energy in the Fight against Climate Change | Magazine Articles | WWF,” 2015. [Online]. Available: https://www.worldwildlife.org/magazine/issues/summer-2015/articles/importance-of-renewable-energy-in-the-fight-against-climate-change--3. [Accessed 20 October 2020].

[19] Arduino, “Arduino UNO R3,” [Online]. Available: https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf. [Accessed 24 August 2022].

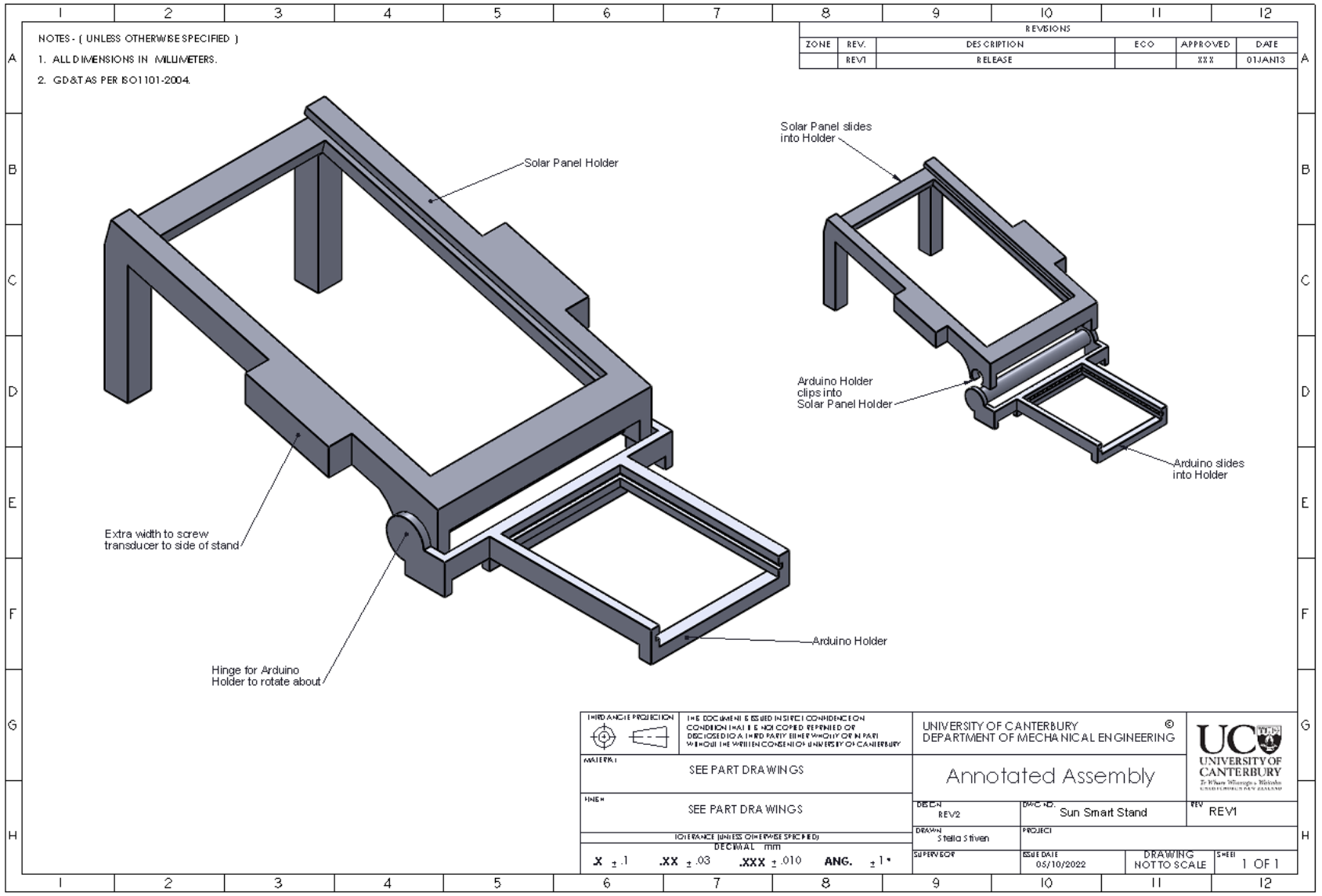
[20] A. McInnes, “PCB Design Rules and Guidelines,” 11 August 2022. [Online]. Available: https://learn.canterbury.ac.nz/mod/page/view.php?id=2007164. [Accessed 24 August 2022].

[21] University of Canterbury, “UC Templates,” 18 August 2022. [Online]. Available: https://www.solidworks.com/. [Accessed 18 August 2022].

Appendix A: Schematic



Appendix B: Stand Annotated



Appendix C: Arduino Code

```
/** @file   uv_sensing_device.ino
    @author Neil Alombro, Bethany Kaye-Blake, Stella Stiven
    @date   3 October 2022
    @brief  Arduino program for solar powered UV sensor. Components include a
            UV sensor, transducer, RGB LED, 7-segment display, and a button.
*/

#include <Adafruit_LTR390.h>
#include <Adafruit_BusIO_Register.h>
#include <Adafruit_I2CDevice.h>
#include <Adafruit_I2CRegister.h>
#include <Adafruit_SPIDevice.h>

// UV Sensor Initialisation
Adafruit_LTR390 ltr = Adafruit_LTR390();
uint32_t uv_data;
uint32_t uv_index;

// Transducer Initialisation
const int buzz_pin = 3;
bool buzz_on = false;

//Time Initialisation
unsigned long overall;
unsigned long minutes_elapsed;
unsigned long time_since_last_buzz;
unsigned long TIME_THRESHOLD = 100;


// RGB Led Initialisation
const int red_pin = 11;
const int green_pin = 10;
const int blue_pin = 9;


// Function that changes the colour displayed by the RGB LED
// Analog writes to the pin with the according ratio in call.
void rgb_color(int red_value, int green_value, int blue_value)
{
  analogWrite(red_pin, red_value);
  analogWrite(green_pin, green_value);
  analogWrite(blue_pin, blue_value);
}

// 7 Segment Display Initialisation
const int a_pin = 6;
const int b_pin = 7;
const int c_pin = 13;
const int d_pin = 12;
const int e_pin = 8;
const int f_pin = 5;
const int g_pin = 4;
```

```
// Multiple helper functions to display different
// numbers and a letter ‘H’ to signify high.
// Writes directly to each pin to set LOW/HIGH.

void clear_display(void)
{
  digitalWrite(a_pin, LOW);
  digitalWrite(b_pin, LOW);
  digitalWrite(c_pin, LOW);
  digitalWrite(d_pin, LOW);
  digitalWrite(e_pin, LOW);
  digitalWrite(f_pin, LOW);
  digitalWrite(g_pin, LOW);
}

void display_zero(void)
{z
  clear_display();
  digitalWrite(a_pin, HIGH);
  digitalWrite(b_pin, HIGH);
  digitalWrite(c_pin, HIGH);
  digitalWrite(d_pin, HIGH);
  digitalWrite(e_pin, HIGH);
  digitalWrite(f_pin, HIGH);
}

void display_one(void)
{
  clear_display();
  digitalWrite(b_pin, HIGH);
  digitalWrite(c_pin, HIGH);
}

void display_two(void)
{
  clear_display();
  digitalWrite(a_pin, HIGH);
  digitalWrite(b_pin, HIGH);
  digitalWrite(g_pin, HIGH);
  digitalWrite(e_pin, HIGH);
  digitalWrite(d_pin, HIGH);
}

void display_three(void)
{
  clear_display();
  digitalWrite(a_pin, HIGH);
  digitalWrite(b_pin, HIGH);
  digitalWrite(c_pin, HIGH);
  digitalWrite(d_pin, HIGH);
  digitalWrite(g_pin, HIGH);
}

void display_four(void)
{
```

```
  clear_display();
  digitalWrite(f_pin, HIGH);
  digitalWrite(g_pin, HIGH);
  digitalWrite(b_pin, HIGH);
  digitalWrite(c_pin, HIGH);
}

void display_five(void)
{
  clear_display();
  digitalWrite(a_pin, HIGH);
  digitalWrite(f_pin, HIGH);
  digitalWrite(g_pin, HIGH);
  digitalWrite(c_pin, HIGH);
  digitalWrite(d_pin, HIGH);
}

void display_six(void)
{
  clear_display();
  digitalWrite(a_pin, HIGH);
  digitalWrite(f_pin, HIGH);
  digitalWrite(e_pin, HIGH);
  digitalWrite(d_pin, HIGH);
  digitalWrite(c_pin, HIGH);
  digitalWrite(g_pin, HIGH);
}

void display_seven(void)
{
  clear_display();
  digitalWrite(a_pin, HIGH);
  digitalWrite(b_pin, HIGH);
  digitalWrite(c_pin, HIGH);
}

void display_eight(void)
{
  clear_display();
  digitalWrite(a_pin, HIGH);
  digitalWrite(b_pin, HIGH);
  digitalWrite(c_pin, HIGH);
  digitalWrite(d_pin, HIGH);
  digitalWrite(e_pin, HIGH);
  digitalWrite(f_pin, HIGH);
  digitalWrite(g_pin, HIGH);
}

void display_nine(void)
{
  clear_display();
  digitalWrite(a_pin, HIGH);
  digitalWrite(f_pin, HIGH);
  digitalWrite(g_pin, HIGH);
  digitalWrite(b_pin, HIGH);
  digitalWrite(c_pin, HIGH);
```

```
}

void display_h(void)
{
  clear_display();
  digitalWrite(b_pin, HIGH);
  digitalWrite(c_pin, HIGH);
  digitalWrite(e_pin, HIGH);
  digitalWrite(f_pin, HIGH);
  digitalWrite(g_pin, HIGH);
}


// Function to display number and colour dependent on UV Index
void display_seg_and_rgb(uint32_t uv_index)
{
  switch (uv_index) {
    case 0:
      display_zero();
      rgb_color(0, 255, 0); // Green
      break;
    case 1:
      display_one();
      rgb_color(0, 255, 0); // Green
      break;
    case 2:
      display_two();
      rgb_color(0, 255, 0); // Green
      break;
    case 3:
      display_three();
      rgb_color(255, 0, 0); // Yellow
      break;
    case 4:
      display_four();
      rgb_color(255, 210, 0); // Yellow
      break;
    case 5:
      display_five();
      rgb_color(255, 210, 0); // Yellow
      break;
    case 6:
      display_six();
      rgb_color(200, 19, 0); // Orange
      break;
    case 7:
      display_seven();
      rgb_color(200, 19, 0); // Orange
      break;
    case 8:
      display_eight();
      rgb_color(255, 0, 1); // Red
      break;
    case 9:
      display_nine();
      rgb_color(255, 0, 1); // Red
```

```
      break;
    case 10:
      display_h();
      rgb_color(255, 0, 1); // Red
      break;
    default:
      display_h();
      rgb_color(127, 0, 255); // Purple
      break;
  }
}

// Button Initialisation
const int button_pin = 2;
int button_state = 0;


// Sets up all components with object instances,
// setting initial values, and setting pin modes.
void setup()
{
  // UV Sensor
  Serial.begin(115200);
  ltr.begin();
  ltr.setMode(LTR390_MODE_UVS);
  ltr.setGain(LTR390_GAIN_3);
  ltr.setResolution(LTR390_RESOLUTION_16BIT);
  ltr.setThresholds(100, 1000);
  ltr.configInterrupt(true, LTR390_MODE_UVS);

  // Transducer
  pinMode(buzz_pin, OUTPUT);
  analogWrite(buzz_pin, 127);
  delay(500);
  analogWrite(buzz_pin, 0);


  //Time
  overall = 0;
  minutes_elapsed = 0;
  time_since_last_buzz = 0;

  // RGB Led
  pinMode(red_pin, OUTPUT);
  pinMode(green_pin, OUTPUT);
  pinMode(blue_pin, OUTPUT);

  // 7 Segment Display
  pinMode(a_pin, OUTPUT);
  pinMode(b_pin, OUTPUT);
  pinMode(c_pin, OUTPUT);
  pinMode(d_pin, OUTPUT);
  pinMode(e_pin, OUTPUT);
  pinMode(f_pin, OUTPUT);
  pinMode(g_pin, OUTPUT);
  clear_display();
```

```
  // Button
  pinMode(button_pin, INPUT);
}

void loop()
{
  // UV Sensor Reading
  // Note: UV Sensor reading is NOT index

  // Time
  overall = millis();

  // Updates UV Index every minute
  if (overall >= (60000 * minutes_elapsed)) {
    minutes_elapsed += 1;
    if (ltr.newDataAvailable()) {
      uv_data = ltr.readUVS();
      uv_index = uv_data / 25;
    } else {
      uv_data = 0;
      uv_index = 0;
    }

    display_seg_and_rgb(uv_index);

    // Add on to sunblock indicator
    time_since_last_buzz += uv_index;
  }

  // Checks if 100 index points have been passed: buzzer turns on if so.
  // Based on research: https://pdskin.com/blogs/uv-index-the-sun-safety-scale
  if (time_since_last_buzz > TIME_THRESHOLD) {
    buzz_on = true;
    analogWrite(buzz_pin, 127);
    rgb_color(255, 0, 1); // Red
    delay(100);
    rgb_color(0, 0, 0); // Blank
  }

  // Button state update
  button_state = digitalRead(button_pin);

  // If buzzing, button will act as a stopper for transducer.
  // Else, button will act as a 7 Segment indicator to show the exact UV Index
  if (buzz_on) {
    if (button_state == HIGH) {
      buzz_on = false;
      analogWrite(buzz_pin, 0);
      time_since_last_buzz = 0;
      display_seg_and_rgb(uv_index);
    }
  } else {
    if (button_state == HIGH) {
      display_seg_and_rgb(uv_index);
    } else {
```

```
      clear_display();
    }
  }
}
```

Appendix D: Final Design