Amun-Ra

Senior Design 2015

Jimmy Blaha

James Heisler

Eleanor Mershon

Chris Short

Table of Contents

Introduction

**Detailed System Requirements**

**Detailed Project Description**

**System Theory of Operation**

## System Block Diagram

## Photo Detection Subsystem

## Rechargeable Battery Subsystem

## Microcontroller Subsystem

#### Sleep Mode

#### Flash Memory

#### Analog to Digital Conversion

## Bluetooth Subsystem

## App Subsystem

**System Integration Testing**

**User Manual and Installation**

**To-Market Design Changes**

**Conclusions**

**Appendices**

# Introduction

**Motivation**

Because of certain society standards on what is considered attractive, most people think tanner is better and that having a tan is considered “healthy.” Unfortunately reality is the opposite where the process to become tan can be deadly. Skin cancer is the most common form of cancer in the US with over 2 million people diagnosed annually. However, it is also the most preventable. There are many different types of skin cancer, including Basal Cell Carcinoma, Melanoma, Kaposi’s Sarcoma, Squamous Cell Carcinoma, and Actinic Keratosis, but the unifying factor is that avoiding overexposure to UV radiation is one of the steps to helping prevent these from developing. According to PreventCancer,org, “protecting the skin during the first 18 years of life can reduce the risk of some types of skin cancer by up to 78%”[[1]](#endnote-1) For this reason, Amun-Ra has worked towards developing a device that keeps users informed of their sun exposure over time and alerts them to when they are reaching dangerous levels.

Although more pigment in one’s skin renders one less susceptible, it is a common myth that dark skinned people will never develop skin cancer. Everyone who does not take the necessary precautions is at risk, not just people with lighter skin. Although some forms of the illness are caused by genetics or other environmental influences, most skin cancer is caused by UV radiation from the sun. Light from the sun is made up of many different forms of radiation including light from the visible, infrared, and ultraviolet spectrums. The rays in the UV range from about 100nm – 400nm have the required energy to mutate human cells and can cause permanent damage to body. While acknowledging that there are some benefits from sun exposure, for example, vitamin D, it is important to note that the benefits from this require only brief exposure (15 minutes per day, 3 days a week).[[2]](#endnote-2) Our project looks to help monitor user’s exposure to hazardous forms of radiation as to allow people to be more aware of the danger and take appropriate percautions.

**How Amun Ra’s device can help**

Many people simply don’t understand the risk they are taking upon themselves by staying out in the sun, especially because the hazard builds up over time. Amun-Ra wanted to create a way for people to easily monitor their prolonged sun exposure so they can see exactly what danger they are in and even have the ability to show recorded data to their dermatologist. Our device is easy to use and portable as to not inconvenience people or discourage them from using it. While we can’t do much about changing people’s ideas of what looks good, we can make them more aware of the danger they are in. Because this form of cancer is not only deadly but also preventable, this device is a welcome addition to the arsenal of detection devices medical professionals can use to keep people happy and healthy.

**User-friendly considerations**

In order to make this device easy to use for consumers, there are a number of factors that were added to the overall design. One of the main constraints was that the device had to be portable. To reach this goal, Amun-Ra powered the photo detection system using a rechargeable battery with enough capacity for at least eight hours of handling. This keeps the device small and ensures that people don’t need to open up the packaging and remove the battery when it runs out. Also, the group wanted to get rid of extra wires outside the case to increase both the durability and portability. To address this, Amun-Ra went wireless, using a Bluetooth connection to transfer data. Finally, another aspect of portability is size. Amun-Ra tried to make the end product as small as possible as to not inconvenience the user and keep the device inconspicuous.

In addition, a big part of being user friendly was incorporating the smartphone application. Most people keep their phones with them all the times and keep all sorts of information on them, from banking data to emails. Connecting the device to an app helps keep all personal information in one place and makes it less likely for people to lose the data over time. Also, the app allowed for each user to differentiate their profile according to their skin tone. This adds robustness to the final design and makes the system applicable to a wider range of people.

Another benefit of having the device communicate with a smartphone app is that while the app is continuously monitoring the total sun exposure, it can alert the user when he or she as reached a dangerous level via sound or pop up notification. This immediate alert allows the user to know in real time when they’re in danger of getting too much UV and gives them the opportunity to rectify the situation by moving to the shade or reapplying sunblock.

**How well did the finished product work?**

There were a number of things that went smoothly through the design and prototyping process and a number of things that introduced problems and had to be corrected. For example, Amun-Ra realized that the Bluetooth was not going to connect with the app as often as hoped, so a flash memory chip was added to ensure that no data is lost even if there isn’t a successful connection or transfer each minute. In order to ensure longer battery life, decreasing the current consumption was necessary. While adding in the sleep mode feature helped, decreasing the clock frequency would also make a difference. Unfortunately, when the team tried to work at 40MHz instead of 80MHz, the Bluetooth would no longer work correctly. Therefore, it was decided that the clock would stay at 80MHz.

Finally, the biggest problem with the project overall was working with the board designed by the team. Instead of using the PIC32MX695 or PIC32MX795, Amun-Ra decided to use the PIC32MX270 because it still had all the necessary features but less pins making it easier to solder and decreasing extra current consumption. Unfortunately, this introduced a number of problems to the project and the team was unable to program the new microcontroller using the MPLAB IDE.

# Detailed System Requirements

 For this project there are a number of overall system requirements to be met before the device can be considered operations. These include some sort of way to measure incident light, a way to process this light intensity information, a way to move the data from the device to the user interface, the user interface, and a way to power the device. Once broken down into systems, the team could then work on each individually and make decisions about the specific parts associated with them. Each of these systems can be broken down into more detailed subsystems, explanations of which are provided in the below sections.

**Measuring Incident Light**

 The first part of the project dealt with how to get usable data onto the device when the user was standing in the sun. There were a number of ways suggested to go about doing this, including using a photo diode with some sort of filtering mechanism to focus on only certain wavelengths of light that are harmful to humans. In the end, the group decided between three main parts, the VEML6070, the ML8511, or the GUVA-S12SD. All these parts include photo detectors that gather incoming light and output a signal containing information about the strength of the light. Originally planning on using the GUVA-S12SD, the group would connect the output of the photo detection circuit to the input of the microcontroller’s A/D converter to then process the data. However, as it turned out that this particular part was very hard to order, the team decided to look into testing both the VEML6070 and the ML8511. With the ML8511, the output will still need to be processed by the microcontroller. Unlike that, the VEML6070 part does it’s own A/D conversion and connects to the microcontroller via and I2C interface. Both these parts would perform the correct function for the project, it remains a matter of which works best with the finished board that the team ultimately decides to go with.

 The photo detector is mounted on the board in such a way that it can see the sun, but not be in risk of getting knocked off. After considering both lenses and light pipes to focus the light from the packaging down onto the board, the group decided to simply mount the photo detector on the board and remove the packaging above the sensor. This helps protect the photo detector while maintaining a level of accuracy needed to provide useful data.

**Processing the Light Information**

 Processing the information about the light intensity begins in the microcontroller after the data is passed from the photo detection circuit. Here, the microcontroller performs A/D conversion and takes an average value over a period of time to remove outliers form the data set. This information is then passed to the user interface for subsequent processing.

**Transferring Data**

 To get the data from the microcontroller to the user interface on the smart phone, the team decided to use Bluetooth. This adds mobility to the device because the user isn’t tethered to anything for data transfer. The part the team decided to go with was the nRF8001 Bluetooth IC by Nordic Semiconductor. This part accepts data from the microcontroller and sends the information to a connected transceiver, like the one in a smartphone. The main reason this part was picked was the fact that it is “low energy.” As stated above, both mobility and powering the device are concerns for this project. By keeping power consumption to a minimum, the team hopes to increase the longevity of the battery in between charges and allow for a more user-friendly product.

**User Interface**

 Another main system requirement for this project is the user interface, a way for users to be aware of how much sun exposure they are getting based on the data collected by the photo detection unit. The team decided to make an iOS app for iPhones to display the data. The main job of the app is to allow the user to say what their skin type is, track their sun exposure over time, and set alerts for when their exposure reaches dangerous levels. Therefore, the app has to be able to accept data from the device using the Bluetooth transceiver on the phone and process the data through an algorithm that converts the light intensity to a readable and graphable quantity users will understand. It also needs to be able to keep track of how much sun the user has been exposed to over a certain period of time so to make use of the alarm feature.

**Powering the Device**

 The main concern with providing power to the device was allowing it to remain portable while not sacrificing battery life. The team decided to implement a rechargeable lithium polymer battery for a number of reasons. First, after taking generic current measurements and doing some research on available batteries it was concluded that the smallest batteries physically that still held enough charge were of the lithium polymer type. Also, this type of battery allowed for it to be rechargeable. This was a key feature the team wanted to implement because it saves the user the hassle of opening the case and replacing the batteries. Incorporating a rechargeable battery was meant to provide ease for the user and also prolong the lifetime of the device itself as the user will not be opening the protective casing.

# System Theory of Operation

**System Theory of Operation**

For the AmunRa project there will be a flow of data through three major components: the photo detection system, the microcontroller, and the smartphone application. The photo detector collects UV information from the current environment using a light sensor. This is then passed to the microcontroller, which converts it from an analog voltage signal to a manipulatible digital representation and stores it in flash memory. Once a significant amount of data has been collected the microcontroller initiates a transfer to the smartphone via Bluetooth. All algorithms and calculations to turn the UV data to a readable and understandable representation happen through the application. The results are displayed in a variety of ways, including graphically and numerically, in an effort to give users a better idea of what is actually happening.

The device will be powered through a (3.3-3.9V) Lithium polymer rechargeable battery. While this keeps the device portable and easy to use, it requires that the system be as efficient as possible to minimize unnecessary parasitic power consumption. The use of flash memory to store data values is a direct result of the desire to decrease energy consumption. Instead of connecting with the Bluetooth transceiver on the phone and sending information every time the environment is sampled (30samples/min), the values will be stored and accumulated until a certain value is reached. Then, the microcontroller will try to pair with the application and send the data. This decreases the number of times the microcontroller is reaching out to the phone.

Similarly, the microcontroller does not need to be running during the time it’s not sampling, converting, storing or transferring the data. Since the group decided to sample the photo detector about every 2 seconds and the converting-storing process is on the order of microseconds, the microcontroller is essentially doing nothing most of the time. Therefore, it enters a sleep mode (one of two power saving modes available on the device), which shuts off unnecessary components of the microcontroller. Here, current consumption decreases to a point where significant power is saved. After approximately 2 seconds the microcontroller will wake from sleep mode and continue with the rest of its cycle.

The application side of the project deals directly with user interface concerns; how can the data be represented in a way that makes sense to users and is helpful in preventing skin damage due to sun. The data is accepted from the microcontroller using the phone’s Bluetooth transceiver. It is then displayed in a way that allows users to keep track of sun exposure over time. This separation of data collection and data display helps keep the device small but does rely on the assumption that a large number of people have smartphones. By utilizing the phone’s memory and ALU, less work needs to be done by the device itself and the cost for parts is kept down.

**System Block Diagram**

**Ultra-Violet Detection Subsystem**

 The UV sensor is crucial to the projects goal of sensing UV to warn of sunburn. As such, the subsystem must be able to reliably detect UV light and transform UV-Index to a voltage. This voltage must then be able to interface with either the Bluetooth (to send it to the iOS app) or the flash memory (or store the data).

 The sensing circuit utilizes the GUVA-S12SD UV photodiode made by Roithner Laser. The UV photodiode’s responsivity is shown in the appendices. The diode responds to wavelength between 240 and 370nm with a peak responsively around 360nm. This is ideal as it captures most of the UV-A spectrum (315-400nm) and the UV-B spectrum (280 to 315nm); additionally, the photodiode “weights” the UV-A spectrum more heavily than the UV-B spectrum, both of which contribute to sunburn.

A transimpedence amp was used to both amplify the current signal and convert it to a voltage that can then be used by the microcontroller’s analog-to-digital block. The transimpedence amplifying circuit schematic can be seen in Figure XX. The transimpedence amplifier utilizes a single rail (no –Vss), so no negative voltage had to be implemented on the final board design, reducing board complexity and reducing overall project size.



*Figure XX. The circuit schematic for the transimpedence amp used in the UV sensor*

 Due to concerns about the availability of the GUVA-S12SD photodiode, another UV sensing option was chosen: Vishay’s VEML 6070. Similar to the analog sensor, the VEML sensor has peak responsivity of 355nm, and a 40nm bandwidth. Unlike the GUVA sensor, the VEML is digital and uses an I2C output. The part itself is only 4.23 mm2, which is ideal for our size constraints; however, the part’s recommended configuration also requires an additional five resistors and one capacitor. The data sheet for the VEML 6070 can be found in the attached appendices.

 When using the analog (GUVA-S12SD based) sensor, the output voltage of the sensor is simply feed to an analog input pin of the microcontroller. An analog-to-digital conversion is later performed on sensor output voltage; this is detailed in the microcontroller subsystem section.

 The Vishay VEML 6070 sensor, which is on the board but not utilized due to time concerns and the previous use of the GUVA-S12SD option, uses an I2C interface to communicate with the microcontroller.

 One of the main engineering decisions was whether to use a digital or analog UV sensor. The analog sensor simply outputs a voltage, while the digital sensors typically have a DSP block on the breakout board. The digital sensors often just output a UV-Index value. Much of the time this value was not derived directly from the UV wavelengths, but rather extrapolated from a visible light sensor using a typical black body curve. This was undesirable, as it would output misleading values on a cloudy day or when inside or shaded areas. In these locations visible light is present, while UV light most likely is not (it is filtered through glass or clouds). For this reason, the team decided to use the analog UV sensor. The particular part, the GUVA-12SD was chosen, as its response curve was limited to UV wavelength, so it would not sense near-UV rays. Additionally, it claims to have an output that is linearly proportional to the UV Index. This allows for more straightforward and reliable conversion from UV index to the minimal erythemal dose (MED). The analog sensor is also physically smaller, which helps address overall product size concerns. While the VEML 6070 has many of the same characteristics, the part could not be used until the board was completed and functional, as the VEML part has an unusual (and very small) package (OPLGA-6). This led the team to decide to utilize the photodiode and transimpedence amp option.

 The analog UV sensor was tested by solar irradiance and comparing it to a Tenmars TM-213 UV A/B light meter. The Tenmars device is designed for measuring UV intensity at the wafer plane, and is very precise. It measures the UV intensity from 290-390nm, which is similar to, but not exactly, the range of the GUVA-S12SD photodiode. The sensor was tested during various conditions (sunny, cloudy, etc. and at varying angles). A table with test values is provided in Table XX. As can be seeing the corresponding graph in Figure XX, the UV index (as measured by the analog-to-digital) is linear with respect to intensity. There are slightly deviations due to sensor viewing angle.

*Table XX. Data when testing the GUVA sensor*

|  |  |  |
| --- | --- | --- |
| Tenmars Intensity Reading (μW/cm2) | A2D Output Voltage | Notes: |
| 880 | 0.26 |  |
| 800 | 0.20 | 45 deg. tilt |
| 1113 | 0.33 |  |
| 1113 | 0.287 |  |
| 1679 | 0.603 |  |
| 1750 | 0.635 |  |
| 1120 | 0.302 | Tilted 45 deg. Away from sun |
| 1550 | 0.355 |  |
| 3200 | 0.547 |  |
| 1300 | 0.725 |  |
| 1260 | 0.523 |  |
| 1250 | 0.474 |  |
| 1250 | 0.413 | Straight up |
| 1250 | 0.690 | Directly at sun |
| 800 | 0.284 | 30 deg. Away from sun |
| 1000 | 0.275 |  |
| 393 | 0.350 |  |
| 420 | 0.110 |  |
| 470 | 0.113 |  |

Figure XX. Test Data from the GUVA-S12SD Sensor.

**Battery Recharging and Power**

 As the UV-Sunburn sensor is designed to be mobile, the device must be able to be used for a full day. The battery must support this roughly eight-hour period of use. After the device is used for the day, it must be ready again the next day. To do this, the design team chose to utilize a rechargeable battery. This will minimize cost for the user, while also allowing for packaging that does not have to open for batteries to be replaced. The recharging circuit must be able to receive power from another source via a connector. Thus, the battery and power module must be able to supply adequate power for an eight-hour period and then be recharged in less than five hours.

 This functionality will be implemented via the circuit seen in Appendix XXX. The recharging function of the circuit comes from the MCP 73831, a charge management controller from MicroChip. The specifications for this part can be found in Appendix XXX. The circuit follows the typical application set forth in the data sheets. It uses two 4.7-microfarad capacitors to provide stability at the input and output of the MCP. Resistor 1 protects the two status LED’s, while the three programming resistors provide options for the preconditioning current value. The charge management controller performs the bulk of the work, stopping the charging when the battery is full and the other charging controls.

 The decision to design a recharging circuit based on the MCP 73831 was based off of the parts small size, low cost, and limited number of peripheral parts (further minimizing space). As physical size is one of the team’s main concerns, the MCP’s ability to perform the desired functionality while occupying the smallest physical size lead the team to choice of the MCP 73831 as the basis for the charging circuit. It also interfaces well with the USB power bus, a big plus as this was the preference for the power source. Mini-USB ports are common, and are often used to charge phone, cameras and similar devices while being a bit more rugged than the more common Micro-USB this led the team to choose the Mini-USB port for charging. While the product would come with a Mini-USB cable, it would also be easily replaceable if lost or damaged.

 The charge management controller chosen was designed to charge the 3.7v Li-Poly battery. Consequently, the team chose a Li-Poly battery, and decided to mount in on the final board via a JST connector, which provides a durable and rugged connection. The next decision was the energy of the battery. Larger batteries may last longer, but also occupy more space and are more expensive. The team used the average current consumptions from the spec sheets of each major block and the relative time each operation would be executed. Doing this, it was computed the battery must contain at least 308.954494 mAh for the device to last eight hours. After efficiencies of the DC-DC converter, battery and charge controller are taken into account, this number would further increase; if all three had efficiencies of 90% this would increase current consumption to roughly 420mAh. While one battery had energy storage of 450mAh, the 850-mAh version was chosen as it provided much more wiggle room at very little price and additional space. The Matlab code used to arrive at the 309 mAh can be found in Appendix XX.

 As the nominal voltage of the battery in 3.7 volts and the microcontroller takes 3.3 volts, a DC-DC charge pump was used to bring the voltage down. The DC-DC should provide a higher efficiency than a charge regulator. The MCP1252 was chosen as the DC-DC charge pump, as it is a commonly available part that fit the requirements.

 The powering circuitry is mostly independent from the functionality of the other modules (outside of the obvious fact that it has to power them). One exception to this is the sleep mode governed by the Watch Dog timer. The sleep mode halts the CPU, so the microcontroller module draws less current. This was done to extend the battery life and ensure the device can function for the desired nine-hour period. More details on the sleep mode can be found in the microcontroller subsystem design.

Appendices:



Charging Schematic

Data Sheets:

Tenmars Link: http://www.tenmars.com/webls-en-us/TM-213.html

Vishay Datasheet: <http://www.vishay.com/docs/84277/veml6070.pdf>

MCP 73831: <http://ww1.microchip.com/downloads/en/DeviceDoc/20001984g.pdf>

MCP 1252: <http://ww1.microchip.com/downloads/en/DeviceDoc/21752B.pdf>

GUVA-S12SD: <http://www.roithner-laser.com/datasheets/pd/uv/guva-s12sd.pdf>

850mAh Li-Poly Battery: <https://www.sparkfun.com/datasheets/Batteries/063048%20Li-polymer.pdf>

**Microprocessor Subsystem (sleep mode, flash memory, A/D conversion)**

### Microcontroller: PIC32MX270F256B

For our microcontroller the Amun-Ra group decided to downsize from a Microchip PIC32MX695F512H to a PIC32MX270F256B. Still a PIC32 series, this new microcontroller has less nonessential features than others and also only has 28 leads. The microcontroller on this device has a limited set of necessary functions; analog to digital conversion, storing data in the memory through SPI transfer, and initiating Bluetooth transmissions are essentially it’s only roles. Choosing a part with fewer extra features helps keep current consumption to a minimum and avoids parasitic power losses. In addition, the small number of leads coming from the device makes it much easier to work with due to the smaller number of contacts to solder. Picking the PIC32MX270F256B specifically out of all the other 28 lead PIC32MX parts was desirable because it has the most program memory. This allows for flexibility when programming and provides an area to optimize when improving the product later on.

This part is rated for 2.3-3.6V, which encompasses the desired operating voltage of 3.3V. It has a sleep mode that can be awakened with a Watch Dog Timer and two SPI’s, one to communicate with the external flash memory and another to talk to the Bluetooth transceiver. It also has a 10 bit ADC to convert the analog signals to digital. This microcontroller was chosen because it has everything needed for the project and minimum extraneous features. While there are other options for programming Microchip PIC devices, AmunRa chose to program this device in C using MPLAB IDE because it was what the group members were most familiar with and was appropriate for the task.

Sleep Mode

(refer to Appendix for code)

As described in the above section, the microcontroller does not need to be drawing current during the times when it is not doing useful work (the clocks don’t need to be running). On the Microchip PIC32MX microcontroller there are multiple power saving modes, including CPU running and CPU halted. There is no reason for the CPU of the microcontroller on this device to be running when it’s not doing anything. Therefore, after a cycle of sampling, converting, and storing it enters into sleep mode, which shuts down both the CPU and the system clock source, limiting the need for current and decreasing power consumption.

The microcontroller is programmed to wake up based on the Watch Dog Timer, which is essentially a counter that counts to about 2 seconds. The WDT uses an NMI (nonmaskable interrupt) to pull the device out of the power saving modes. The NMI is similar to a device reset, but registers and peripherals are not reset and the software picks up in the code at the point after where the sleep mode was set.

The microcontroller for the Amun-Ra project works on a loop. It samples the photo detector, performs analog to digital conversion, and stores the value in an array. After about a minute of sampling (24 samples) the microcontroller takes the array of collected values, disregars the highest three as outliers and averages the next highest five samples as a representative value for the user’s sun exposure over that minute. At this point, the microcontroller will remain awake long enough to attempt to connect with the Bluetooth on the smartphone and transfer the stored data. If a successful connection is not made, the microcontroller will store the average value in the flash memory and continue with its sleep/sample process for another minute before it attempts another Bluetooth connection and transfer. This process allows the microcontroller to remain in sleep mode for most of its operation, therefore maximizing the efficiency of the system and keeping power consumption to a minimum.

In order to test the Sleep Mode portion of the code the group wrote a separate program the blinked one LED twice, then went to sleep for 8 seconds, woke up and blinked the LED twice again. By hooking the kit board up to a power supply, the group measured the current when the microcontroller was awake and when it was asleep. It was found that putting the device into sleep decreased the current consumption from about 84mA to 45mA. While these numbers are specific to the kit board used for testing, it shows that entering sleep mode helps decrease power consumption significantly and gives the group some numbers to work with when choosing a battery.

Flash memory

(refer to Appendix for code and spec sheet)

For the Amun-Ra project, the need for some on device memory (as opposed to relying on smartphone memory) was necessary given that data transfer via Bluetooth does not occur every time the environment is sampled and therefore there are extra data values. Flash memory is used because it is block erasable, which was decided to be the most appropriate for the required data storage. Once the information is transferred from the memory to the phone, it is no longer beneficial to keep on the device and can all be erased. Erasing blocks at a time speeds up the process of removing the obsolete data and provides an easy path to clearing the way for new information.

The flash memory used in this project is a Silicon Storage Technology 16 Mbit SPI serial flash, SST25VF016B. The voltage range is from 2.7V to 3.6V and current consumption during a read is around 10mA. There are two ways to write data to memory on this part. The AnumRa group chose to do a byte write instead of Auto Address Increment Word-Program because each UV sample only requires 8 bits of resolution; only one byte needs to be written at a time. This adds some complexity because in order to ensure that the byte is actually stored in the memory, the address of the memory location has to by empty before the write, and manually incremented after. By checking to see if an address is empty and keeping track of each write, the group was able to ensure that no data was lost by overwriting.

Similarly, keeping track of where the last data value was written is essential to making sure all the data is read. The group used a function that keeps reading up until the address of the last byte write. Once this data is read it can be sent via Bluetooth to the app. After a successful Bluetooth pairing and data transfer the memory is fully erased and the address incrimination starts over from zero. If the phone’s Bluetooth is not within range or a connection is not created, the data will remain in the flash and samples will continue to be converted and stored until there is a successful connection. There is also a feature on the SST25VF016B that allows for block protection, essentially protecting parts of the memory from being erased even during a full chip erase. The Amun-Ra group didn’t use this feature since, as previously stated, once the data is transferred to the app it is no longer needed on the device and can be erased. However, if extra components were later added this feature could be useful.

To test the flash memory, the group made use of the logic analyzer. With the logic analyzer connected to the correct pins (MOSI, MISO, Chip Enable, and CLK), it was a matter of writing to a specific memory address, waiting until the memory write was completed, then reading from that same address. Watching the output of the logic analyzer, the Amun- Ra group could see that data values were being written and read from the correct addresses. They also tested that the address incrementation worked by writing 100 values to successive addresses in the flash and then reading back the last few addresses. This showed that the software was putting successive data values in the correct places and the reading back the same values.

Analog to Digital Conversion

(refer to Appendix for code)

The data being passed to the microcontroller from the photo detector is in the form of an analog signal. In order to store, transfer, and manipulate this information it is necessary to convert it to a digital representation. To do this the group made use of the ADC built into the PIC32MX270F256B microcontroller. The microcontroller samples the photo detector as an analog input, and then converts it to a 10 bit digital representation in the form of the original analog signal. Since the flash memory is stored in 8 bit (one byte) addresses, this value will lose resolution as it is saved. However, 8 bits of resolution was determined to be precise enough for later analysis in the app stage.

The ADC on the microcontroller connects to the input pin and samples for a time called “acquisition.” After a sufficient acquisition time has been reached, the stable analog voltage value is stored in a “sample and hold amplifier” (SHA), and the input pin is electrically disconnected from the microcontroller. Now the conversion process starts. Conversions require 12 TAD cycles, where TAD is the period of the ADC clock and is can be set in the software based off of the peripheral bus clock. TAD for this device is about 200ns, so total conversion time is about 2.4us. After conversion, the data is a 10 bit floating point value. These 10 bits are then converted into an unsigned integer two orders of magnitude larger than the original value (for a voltage swing of 0-1V output from the photo detector that values sent to the app range from 0-100). Converting from floating point type to unsigned int type loses some precision, but still keeps the integrity of the value enough to be useful as a data point.

Testing the ADC was a matter of hooking up a potentiometer to the input pin of the microcontroller’s ADC. By giving the potentiometer a voltage range from 0-1V (matching that of the output of the photo detector), the group could show that all those analog voltages were converted to digital values and then displayed on the LCD.



Microcontroller Flow Chart 1 Flow Chart for the Microcontroller Showing its Operation

***Bluetooth Subsystem***

 We used Nordic Semiconductor’s low energy Bluetooth transceiver component nRF8001 to achieve the Bluetooth communication between the board and the cell phone application. Nordic Semiconductor provided the framework for the programming via the open source development kits which allow the program to be adapted and developed without licensing. In addition, the Bluetooth protocol is able to be used cross-platform and has been demonstrated on both iPhone iOS and Samsung Galaxy Android. Early on, the use of the board allowed us to communicate with a Bluetooth radio on a phone easily because the Nordic had apps on both platforms’ app stores that used serial communication between the board and phone. Therefore, the largest challenge to overcome was adapting the microprocessor to the code to actually implement the routine that was provided.

*Physical Chip*

 We interface the nRF8001 to the microprocessor using an SPI interface which will transmit the digital information measured by the photodiode. There is also on chip memory which will allow some flexibility in case the receiver is temporarily out of range from the chip and data cannot be transmitted to it. The current consumption is less 14mA while connected and as low as a few hundred nA when not connected. There is a DC/DC converter that allows the current to drop even further, potentially by 20%. The on board clock oscillates at 16 MHz.

*Subsystem Requirements/Testing Results*

 The Bluetooth radio has several requirements that it must be able to meet. First, the Bluetooth cannot drain too much battery to make the long period of use not feasible. Nordic Semiconductor made this chip specifically for these applications so it is no surprise that the initial testing indicates that the power consumption is within the acceptable range. Also, the required supply voltage must be able to be sourced by the microcontroller. The nRF8001 requires a source voltage of 1.9-3.6 V which is able to be supplied by the PICkit quite easily. Next, there does not have to be a lot of packets delivered but there is a limit on the bytes of the delivered package. A single delivery is limited to 20 bytes so the data transmitted must not exceed this size. The precision available for 20 bytes of voltage data is acceptable for our applications as it exceeds the realistic precision of the photodiode itself. Lastly, the Bluetooth has to maintain a reliable connection at relatively short distances. While this is not easily quantified, through our testing, the chip has maintained steady connections for long durations even at distances of a few yards. In conclusion, the requirements have been met through the testing of this component and will be able to interface the board to the cellular device which has already been demonstrated.

*Software Overview*

 The interface the Bluetooth communicates in is the universal asynchronous receiver/transmitter (UART) so the initial setup for communication is establishing a baud rate. Instead of initializing to the baud rate that was standard for the nRF8001, we establish a new rate of 57600 Hz for use with the PICkit. Next, the setup protocol for the nRF must be loaded onto the board; essentially, this is called the Application Controller Interface which is a proprietary layout that must be implemented on the nRF8001 for use.

 Once the setup has been completed successfully, the processor enters into the steady looping state of the program. The ACI must update through any iteration and execute any commands that have been transitioned into the communication port. This occurs by first checking if there exists a command and exiting if not. Next, it is necessary to analyze the nature of the event with a case specific execution pattern that follows as:

Case1 – ACI Device Started – this occurs when the device is in standby or is just started up. Again a case specific routine is called upon where the processor raises the startup\_required bit if the device is in startup or initiates attempted Bluetooth pairing.

Case2 - command response event comes with an error – handles errors in the response from a Bluetooth transmission. Meaning data transmitted but an error code returned instead of a successful message sent.

Case3 – device connected – Bluetooth initiation is activated

Case4 – pipe status – this checks to see if the terminal is available for read/write. Then, bits are sent into the terminal if it is available or stored if not.

Case5 – event timing changed – to prevent cut off errors, the expected time of each event can be modified to anticipate strings of arbitrary size.

Case6 – event disconnected/advertising timed out – whether due to inactivity or component being turned off, the device has lost connection. The advertising is restarted and the board becomes available for connections again.

Case7 – event data received – the first check in this routine is to find out whether or not the entire data stream is being read in. If so, it is a simple transfer of the UART to print onto the terminal. If not, the routine is started for splicing together packets into coherent data. This is handled by a process control point that alerts the next iteration that the event timing has been changed and tracks the start and end byte of the transmission.

Case8 – data credit event – a credit available is added.

Case9 – pipe error – increments the credit available because a packet was not able to be transmitted. This is also triggered when an incorrect protocol is experienced.

Case10 – hardware error – the program has experienced a hardware error and the line that the error occurred at is printed to the user. Advertising the connection is restarted.

 Getting back into the steady state loop, the UART is cleared and the serial port is read from.

 Protocol is mainly maintained by valid advertising and accepting of radio signals. The packet protocol limits the size of each packet to 20 bytes as well. Moving out of range or remaining inactive for too long also constitutes a disconnection. The protocol is maintained by the setup function of the code which polls the available connections and ensuring that the baud rate is acceptable on both devices. The stack is follows the Bluetooth Core specification v4.0 by using the following parts:

• Part A: Logical Link Control and Protocol

• Part C: Generic Access Profile (GAP)

• Part F: Attribute Protocol (ATT)

• Part G: Generic Attribute Profile (GATT)

• Part H: Security Manager (SM)

*Implementation*

 In order to make use of this protocol, the implementation needs to break up the functionality into two distinct functions: connecting and transmitting. In order to do this, there is a structure that the aci\_loop() function is going to return. It is going to contain two key Boolean flags to signify whether execution has successfully connected or sent data. These flags allow an if statement to break from the connecting phase once it has been executed and then call the transfer function as many times as necessary to send data to the receiver. This also allows an important functionality for our project – the system doesn’t break when the phone isn’t within Bluetooth range. The system advertises for about 10 seconds before it triggers an “unable to connect” execution which sends the data to memory instead of trying to transfer it when this wouldn’t be possible. This is an important function that protects the system from getting stuck in an infinite loop trying to connect which would result in a number of poor results – running the battery down and not taking any further samples.

# System Integration Testing

 To test the system integration, the team started small, testing each subsystem individually before coming slowly building them on top of each other. This permitted that any problems could be more easily identified and fixed without getting lost in the complexity of the system as a whole. The first two parts integrated together were the photo detector and the microcontroller. The idea was to connect the analog output of the photo detector to the input pins of the microcontroller associated with A/D conversion. Then, the value was printed onto the LCD screen as a debugging method. Using a DMM to check the output of the photo detector and a separate light intensity meter the group was able to confirm that the correct values were being read and converted in the microcontroller.

 Once this was matched, next came adding in the memory storage. Using the logic analyzer for this step was extremely helpful for debugging. Because the flash chip connects to the microcontroller via and SPI connection, the logic analyzer could easily decode MOSI, MISO, CE and SCK. This allowed the group to test a variety of functions necessary to the final design including reading a byte, writing a byte, and erasing the chip. Watching the data on the logic analyzer also helped to give a general idea of how long each function would take, which then determined the number of samples per minute the microcontroller would read in from the output of the photo detector.

 Next the group added in the power saving function on the microcontroller called “sleep mode.” Putting the microcontroller to sleep for two seconds in between each sample and the writing the sample into the flash memory could be tracked on the logic analyzer. This allowed the group to ensure that sleep mode was working properly. At that point the photo detector, the A/D conversion, the flash memory read/write/erase and the sleep mode functions were all confirmed to be working properly, meaning that everything on the device side was integrated together.

 The main integration component for the system as a whole is the Bluetooth. The Bluetooth transfer brings the data from device to the app. This proved hard to test while prototyping because the board being used had only port to connect SPI devices to with the correct adaptor built on, and this project required two, one for the flash and one for the Bluetooth. To solve this problem, the group modified the output port that was designed to connect to the LCD screen. This allowed for the flash and the Bluetooth to be connected simultaneously and tested at the same time.

 Testing the app with the Bluetooth was first done separately using the “Hello World” program the group worked on in the fall. This allowed for testing the app with a program that incorporated Bluetooth, which was known to work properly. It allowed for parallel efforts to be made on the app and on the Bluetooth without one having to wait for the other was finished. Eventually, when all separate subsystems were completed, the group used the modified Bluetooth program to transfer data between the device and the app.

 Finally, the group needed to find someway to test that the battery that had been selected could power the device for at least the eight hours discussed in the system requirements sections. To do this, they looked into how much current is consumed during each part of the device operation. By multiplying the peak current of each operation by the time duration of such, a general idea for the power consumption can be found. The spec sheets for the flash and the Bluetooth devices provide an idea of how much current in consumed during each operation. For the flash, p. 23 of the spec sheet shows that the current consumption during a read is about an average of 15mA, the current consumed during a program/erase is 30mA, and the idle current is about 20uA. Then, looking at the logic analyzer while running a read/write/erase program showed how long each operation took: about 34.5ms for a full chip erase, 63us for a write, and 40us for a read.

 For the microcontroller the team used a circuit board with a similar microcontroller that did not use a linear regulator to scale done the input power, eliminating unnecessary current drain. By using an external DC power supply to power the board and running a sleep mode program, the group was able to see that in sleep mode the microcontroller consumes about 22mA of current and each LED consumes about 4mA. Going off the assumption that the microcontroller will be asleep most of the time, this allowed for a large part of the power draw to be calculated.

 Finally, the Bluetooth power consumption needed to be accounted for. In the nRF8001 spec sheet there is a section titled “Dynamic Power Consumption” (Section 13). Here, the activity of the Bluetooth device is split into two operations: advertising and connection. "The connection and advertising events consist of a sequence of radio transmissions, each of which has individual current drain. The average power consumption of an event is calculated by integrating the current drain over the duration of the event” (39). There are two graphs that the team used to extrapolate how much current the Bluetooth operations would consume:



 The second graph shows that the peak current consumption for an advertising event (with DC/DC converter *not* active) is 0.2mA and the peak current for a connection event (with DC/DC converter *not* active) is 0.1mA. Assuming worst-case scenarios for both of these events gives a time interval of 10 seconds (occurring once every minute) and uses the peak current as the current consumption over the entire interval.

 Clearly there were many assumptions made in the current consumption levels and the operation time interval for the different parts of the system, however this was necessary since the team couldn’t test their board until it would be too late to order a battery. In an effort to overestimate the total power draw, the group decided to use worst-case levels and overestimate as much as possible. Using Matlab, the team showed that total power consumption for 8 hours of device operation comes in around 310 mAh, under the level of the 400 mAh battery. Therefore, it was decided that this battery stored enough charge to power the device for the desired level of time and would be acceptable to use in the final project. See the appendix for Matlab simulation.

# User Manual

***Installation***

The only installation required for this product is the downloading of the Amun Ra iOS application from the iTunes app store. Once the app is downloaded and you have turned on power to the device, you can begin using this product.

***Setup***

Setup for this product includes proper placement of the device. The device works best when it is completely in the sun with no angle of incident. Before taking the device outside, please plug in a mini USB to the charging port, and select the switch to charging. From there ,wait until the battery is fully charged. The device can be on independent of it if it charging or not. If the user decides to go outside, simply turn on the device and place it in the sunlight so that it is not obstructed by any other object.

***Working***

The user can tell if the product is working by looking at the power indicator LED and comparing it to the iOS app. If the light is on and a Bluetooth connection has been made, the product is working.

***Troubleshooting***

If an error occurs during usage, simply reset the device by powering it off, waiting 10 seconds, and powering it back on. Likewise, if an error occurs in the application, hit the reset button in the lower right hand corner of the graphical display.

**To Market Considerations**

One of the biggest market considerations is size and packaging. If we were to move forward with this product, it would be in the team’s best interest to shrink the product and adapt the packaging. One idea was to make the product compatible with an iPhone’s external case. This idea would include creating the packaging so that it also served as an iPhone case. This way, the user would not have to worry about two separate devices, but one integrated product.

 Another market consideration would be hiring a graphic designer to make the application more visibly appealing. The graphic designer could create artistic displays and animations that made the application more enticing for children and parents.

 The final market consideration is UV detection. Our current sensor truly only works with the proper angle of incidence to the sun. For future considerations, it would be ideal to design a way to funnel the light into the sensor so that as the sun moves across the sky, the user would not have to adjust the angle the sensor was pointing.

**Conclusion**

 The utility of the Amun-Ra System is unquestioned. The need for a reliable system to monitor sun exposure and quantify it easily to a user – the statistics for skin disease demand an answer. The Amun-Ra Solutions team showed how an idealized version of the prototype is possible and could be a valid solution to the issues first raised in the introduction of this report. The major room for improvement is in the packaging and in the circuit minimizing. Creating something that is more easily worn, that can last longer, has a little more integration with other pairing devices as well as having an expert in skin damage be able to optimize the algorithm for skin damage. However, the prototype stands as a proof of concept that a portable, battery powered, and Bluetooth enabled system can perform the functions necessary for a commercial device.

 The major difficulties that our group experienced were expanding our knowledge of developing software, both for iOS and microcontrollers; designing a board on Eagle that successfully implements our circuit components; and bringing together multiple complex subsystems into a coherent project. The limitation of our demo kitboard comes from the fact that it only has a limited number of SPI ports which we were using memory, Bluetooth, and UV index circuit meaning that not all could be tested at the same time. The result was independent projects that properly implemented their own functionality but sometimes had difficulty with compatibility when they were combined. The final takeaway is that although this is a viable solution for UV monitoring, it must be made into something that people will want to make use of. Marketing and awareness of the dangers of skin disease will have a major impact on whether this technology could ever be a viable consumer product.

1. http://preventcancer.org/prevention/preventable-cancers/skin-cancer/ [↑](#endnote-ref-1)
2. http://preventcancer.org/prevention/preventable-cancers/skin-cancer/ [↑](#endnote-ref-2)