Microcontroller Controlled Radio Trigger

Alex Nathanson | www.alexnathanson.com | alex@alexnathanson.com
1/4/2017

This project was originally designed as a wireless security alarm for a toilet seat (a surreal work assignment). The idea was that the alarm would go off whenever the toilet seat was lifted up. It's based on hacking a radio trigger for a camera flash. The goal was to make the transmitter element as small, energy efficient, and water resistant as possible, while still allowing the battery to be changed easily.

This documentation is mostly complete, but I anticipate a future update to clarify certain parts, document the housing, and add better images, among other changes. The final version will likely be running on a Beagle Bone Black, instead of an Arduino Uno and a computer on the receiver side.

Contents

Microcontroller Controlled Radio Trigger

Alex Nathanson | www.alexnathanson.com | alex@alexnathanson.com
1/4/2017

Possible Future Changes and Additions

- Remove unnecessary LEDs from trigger section to conserve battery.
- Use DC barrel jack instead of screw terminals
- Determine the behavior - should it ring once? Should it keep ringing until the seat is down again? etc.
- Documentation of housing and switch mechanism will be added as it is developed.
- Add an on/off switch to the transmitter battery.

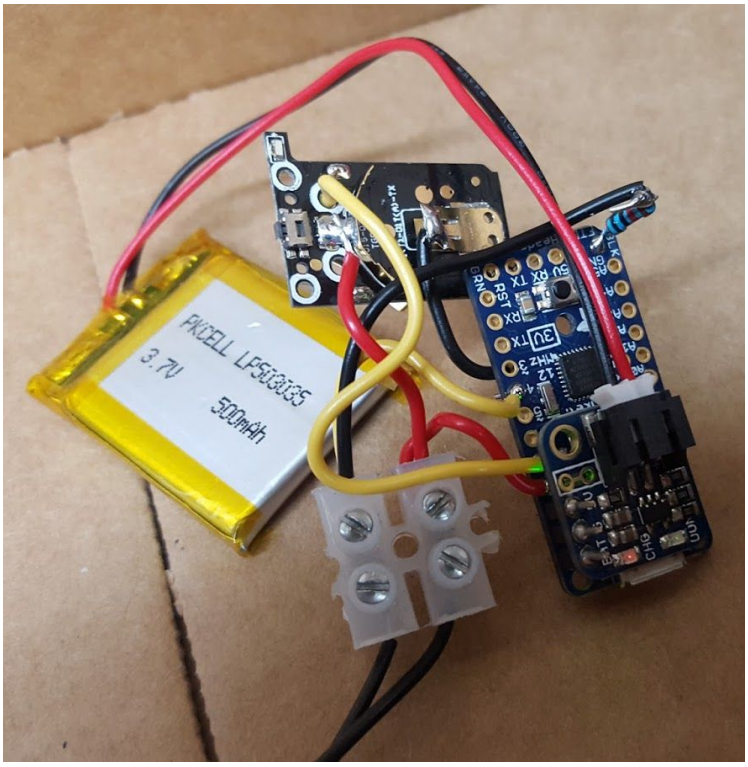
Photo Flash Radio Trigger

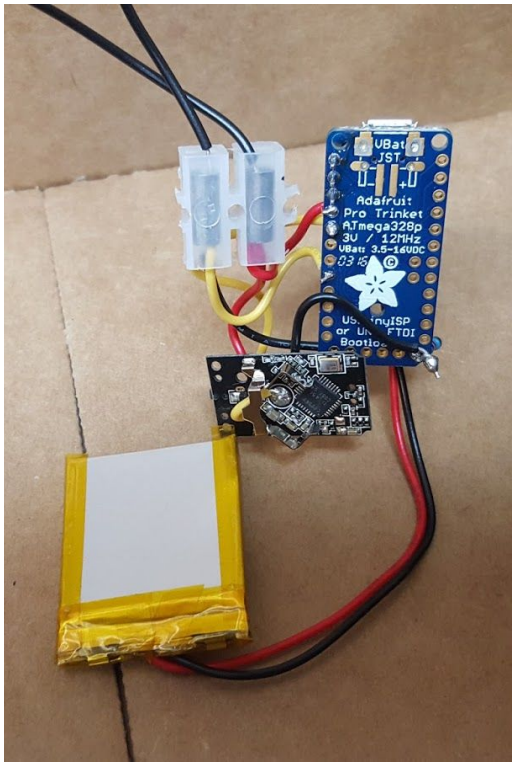
- This project uses a Vello FreeWave Mini-Stand Flash Trigger Set. Any non-TTL paired photo flash radio trigger set will work. If both elements are not designed for +3VDC power supplies, adjust as needed.
- Both the transmitter and receiver of the radio trigger work by connecting the non-grounded trigger wire to ground. By connecting to ground, it was originally intended to short a capacitor in the flash, causing it to discharge. Both elements have 3 important contacts: +3VDC, ground, and trigger. The primary, non-TTL, contacts of a hot shoe are trigger and ground. The ground wire from the power supply can be wired into either the ground contact on the hotshoe or the battery compartment. The transmitter is triggered by connecting the trigger wire to ground. Similarly, when the receiver receives a radio pulse, it connects the trigger wire to ground.

Microcontroller Controlled Radio Trigger

Alex Nathanson | www.alexnathanson.com | alex@alexnathanson.com
1/4/2017

Transmitter





Parts

- Adafruit Pro Trinket 3V
- Adafruit Pro Trinket Lilon/LiPoly Backpack
- Lithium Ion Polymer Battery - 3.7v 500mAh
- 2x screw terminal
- 10K Ohm resistor
- 22 gauge solid core wire
- 3V Radio Trigger
- Electrical contacts for button
- Insulating material for button
- Adhesive for button

Microcontroller Controlled Radio Trigger

Alex Nathanson | www.alexnathanson.com | alex@alexnathanson.com
1/4/2017

Transmitter Power Consumption

Power supply: 500mAh Li-Po Battery

Adafruit Trinket
3.3V input
Amperage draw of Trinket (including LED): 9mA
Amperage draw of “on” button: .3mA
Amperage of LED = 3mA
Total microcontroller draw with LED: 9.3mA
Total microcontroller draw without LED: 6.3mA

Amperage draw of Vello radio trigger
Vello Transmitter User Manual: https://static.bhphotovideo.com/lit_files/127128.pdf
Input Voltage: 3V
Original Power Supply: CR2032
Battery life (in standby mode) with CR2032: 5yr
Typical CR2032 capacity: 220-240mAh
Estimated radio trigger draw (in standby mode): negligible
Estimated radio trigger draw (when triggered): slightly less negligible, but still negligible
Likely will work and draw even less power if the LED is removed. However, sometimes LEDs are needed as diodes as well as feedback to the user. Confirm before removing.
Total transmitter draw: negligible

Amperage draw of battery back pack
TBD

Total amperage draw without removing LEDs: ~9.3

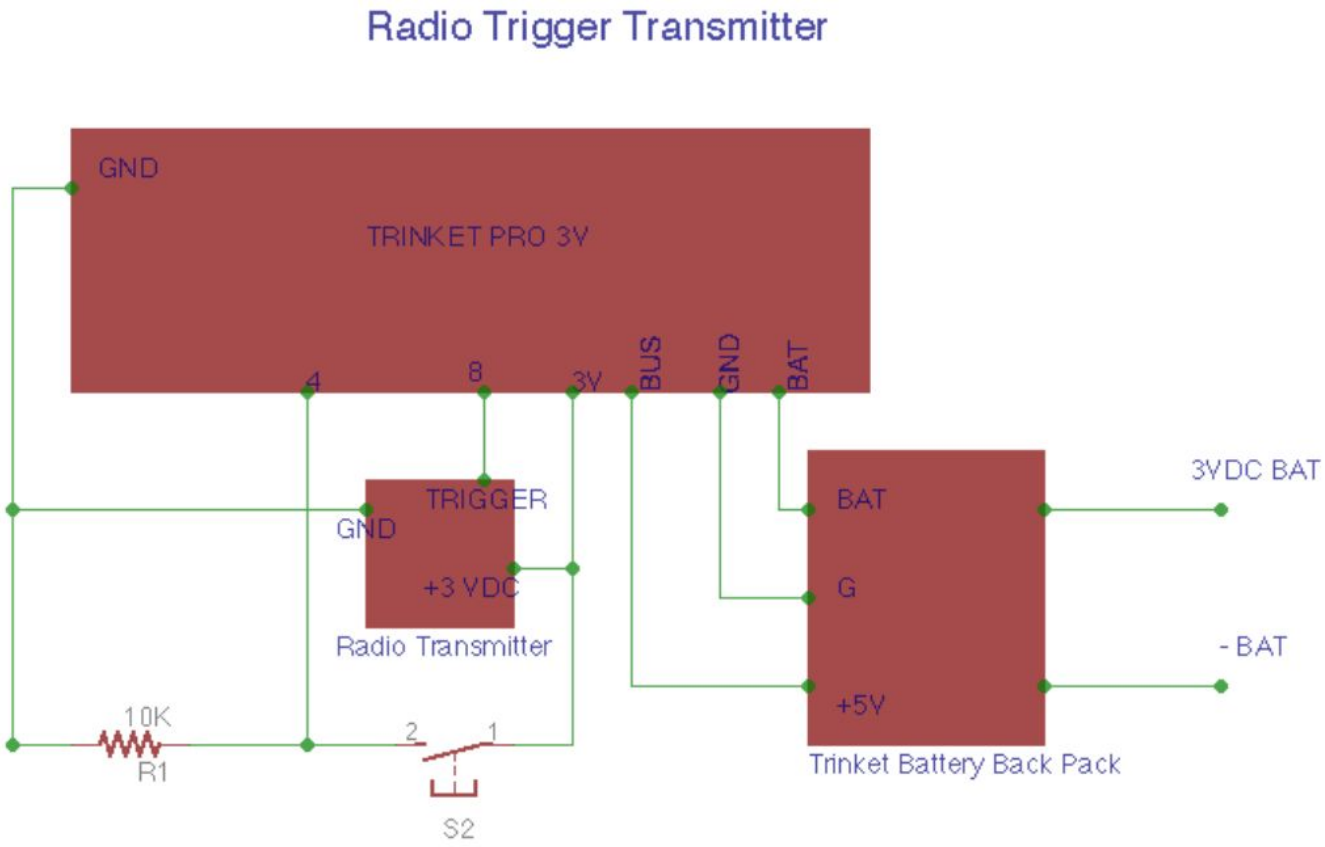
Battery Duration
500mAh/9.3mA= 54h
500mAh/6.3mA=79h

Wiring

Note: The default position is that the switch is closed. When the button is released (in this instance that means, when the toilet seat is lifted) it breaks the connection and triggers the alarm. Pin 8 (trigPin in the code) is set as an output. When it is set to LOW it is connected to ground. This is what enables the trigger to not use a relay.

Microcontroller Controlled Radio Trigger

Alex Nathanson | www.alexnathanson.com | alex@alexnathanson.com
1/4/2017



Notes on the code

This is specific to Mac. Windows requires a specific driver available from Adafruit.

To upload code to the Pro Trinket via USB on a Mac you must add in the correct board manager.

Ardiuno > Preferences > Additional Boards Manager URLs
https://adafruit.github.io/arduino-board-index/package_adafruit_index.json

Currently, the code includes triggering an LED. This is for testing purposes only. It must be turn off to conserve power.

Settings
Board: Adafruit Pro Trinket 3V/ 12MHz (USB)
Port: dependent on computer
Programmer: USBTinyISP

Microcontroller Controlled Radio Trigger

Alex Nathanson | www.alexnathanson.com | alex@alexnathanson.com
1/4/2017

Adafruit Trinket Code 12/24/16

```
/*


 The circuit:

 */

// set pin numbers:
const int buttonPin = 4                                ;     // the number of the pushbutton pin

//for testing only
const int ledPin =  13;      // the number of the LED pin

const int trigPin = 8; //the trigger pin

// variables will change:
int buttonState = 0;         // variable for reading the pushbutton status
int flag = 0; //track button position

void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
  pinMode(trigPin, OUTPUT);
}

void loop() {
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed. If it is, the buttonState is HIGH:

  if (buttonState == HIGH) {
    // turn LED off:
```

Microcontroller Controlled Radio Trigger

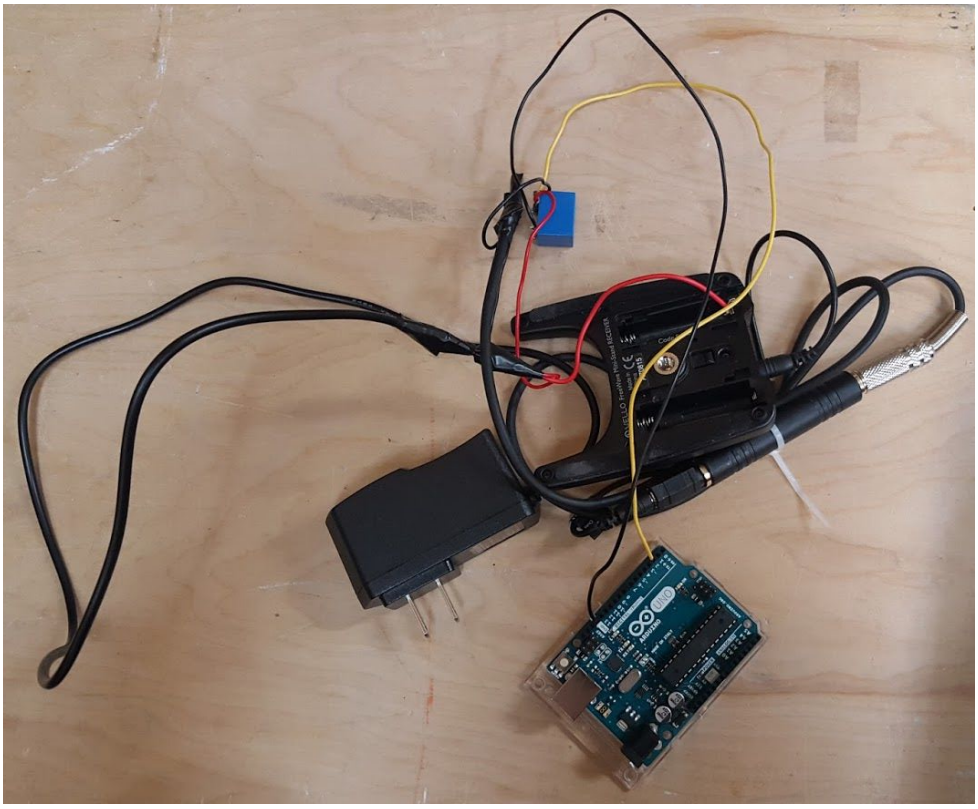
Alex Nathanson | www.alexnathanson.com | alex@alexnathanson.com
1/4/2017

```
    digitalWrite(ledPin, LOW);

    if (flag == 1) {
      digitalWrite(trigPin, LOW);
        delay(25);
      digitalWrite(trigPin, HIGH);
       flag = 0;
    }

  } else {
    // turn LED on:

     if (flag == 0){
        digitalWrite(trigPin, LOW);
        delay(25);
        digitalWrite(trigPin, HIGH);

        flag = 1;

        digitalWrite(ledPin, HIGH);
    } else {
          digitalWrite(ledPin, LOW);

    }
  }
}
```

Microcontroller Controlled Radio Trigger

Alex Nathanson | www.alexnathanson.com | alex@alexnathanson.com
1/4/2017

Receiver



Parts

- 3VDC AC wall adapter
- 3VDC Radio Receiver
- 3VDC relay
- Diode
- Arduino Uno
- USB cable
- Various mono ⅛ or ¼ adapters (optional)
- 22 gauge solid core wire

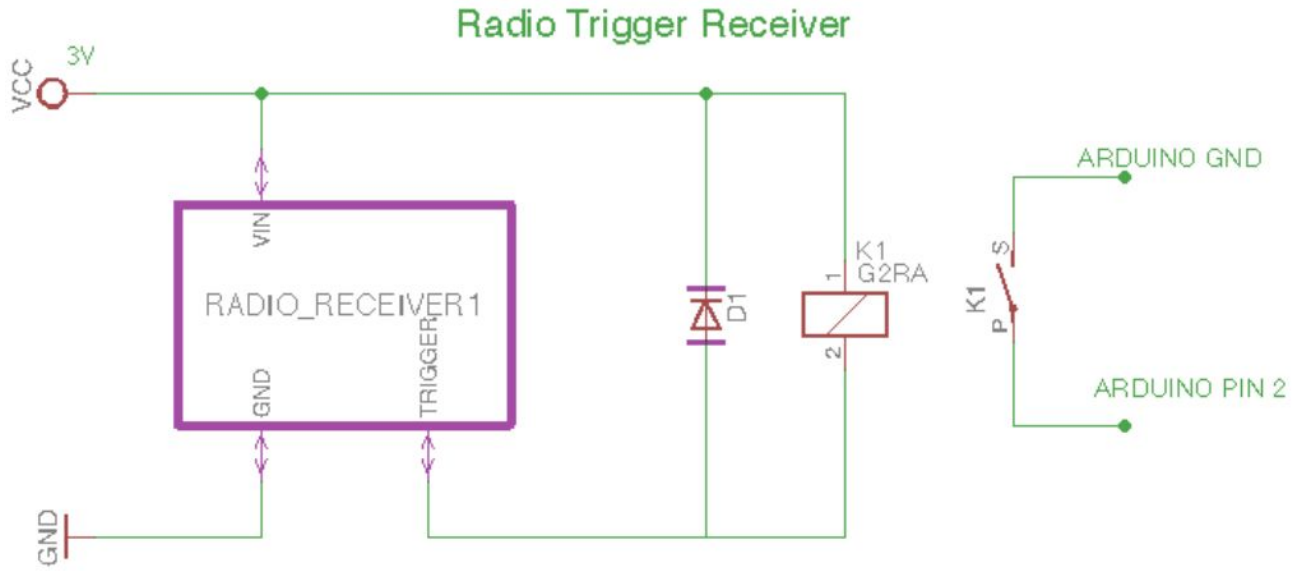
Notes

- Pin 2 (buttonPin in the code) is set as INPUT_PULLUP. This allows the voltage to be constant without an additional 10k Ohm resistor.

Microcontroller Controlled Radio Trigger

Alex Nathanson | www.alexnathanson.com | alex@alexnathanson.com
1/4/2017

Wiring



Arduino Uno Code 12/21/16

```
/*
  TOILET ALARM RECEIVER

 The circuit:
 * LED attached from pin 13 to ground (optional)
 * relay attached to pin 2 from GND

Button code:
 http://www.arduino.cc/en/Tutorial/Button

Arduino to processing code:
https://learn.sparkfun.com/tutorials/connecting-arduino-to-processing
 */


// constants won't change. They're used here to
// set pin numbers:
const int buttonPin = 2;     // the number of the pushbutton pin
const int ledPin =  13;      // the number of the LED pin

// variables will change:
```

Microcontroller Controlled Radio Trigger

Alex Nathanson | www.alexnathanson.com | alex@alexnathanson.com
1/4/2017

```
int buttonState = 0;         // variable for reading the pushbutton status

void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT_PULLUP);


//initialize serial communications at a 9600 baud rate
Serial.begin(9600);

}

void loop() {
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed.
  // if it is, the buttonState is LOW because the pullup resistor is activated:
  if (buttonState == LOW) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);

//send signal to computer, over the serial port
Serial.println(buttonState);
//Serial.println("test2");

// wait half a second before checking sensor. Maybe not necessary because the relay will only send
once...
delay(500);

  } else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
//Serial.println(buttonState);
}
```

Microcontroller Controlled Radio Trigger

Alex Nathanson | www.alexnathanson.com | alex@alexnathanson.com
1/4/2017

Alarm

Parts

- Mac Mini
- Audio Output Device

Notes

- Make sure the file path for the audio file is correct.
- Make sure the port number is correct.

Processing Code 12/29/16

```
// Serial code adapted from here:
// https://learn.sparkfun.com/tutorials/connecting-arduino-to-processing/shaking-hands-part-2

//atomic alarm sound from https://archive.org/details/AtomAlarmSireneNuclearAlarm
  // alarm sound from https://archive.org/download/alarm_162

import processing.serial.*;
import processing.sound.*;

// serial stuff
Serial myPort;  // Create object from Serial class
String val;     // Data received from the serial port
boolean firstContact = false;

// sound stuff
SoundFile file;
Float soundLen;
Integer soundInt;

//variables needed for recording data
PrintWriter output;

AlarmTime myTime; //create object from AlarmTime class
```

Microcontroller Controlled Radio Trigger

Alex Nathanson | www.alexnathanson.com | alex@alexnathanson.com
1/4/2017

```
String seatState;


int seatVal;

// file info
String fileInfo;
String fileStuff;

void setup()
{
  size (200, 200);
  // The first port in the serial list on my mac is Serial.list()[0].
  // Open whatever port is the one you're using.
  myPort = new Serial(this, Serial.list()[1], 9600); //change the 0 to a 1 or 2 etc. to match your port
  myPort.bufferUntil('\n');

  // Load a soundfile from the /data folder of the sketch
    file = new SoundFile(this, "alarm.mp3");

  //set the length of the loop
  soundLen = file.duration();
  soundInt = int(soundLen * 1000);
  // cuts sound before tail of clip to minimize delays. may not be necessary with other clips.
  soundInt = soundInt - 1800;

  myTime = new AlarmTime();
}

void draw()
{
}

void serialEvent( Serial myPort) {
  //put the incoming data into a String -
  //the '\n' is our end delimiter indicating the end of a complete packet
  val = myPort.readStringUntil('\n');
  //make sure our data isn't empty before continuing
  if (val != null) {
    //trim whitespace and formatting characters (like carriage return)
    val = trim(val);
```

Microcontroller Controlled Radio Trigger

Alex Nathanson | www.alexnathanson.com | alex@alexnathanson.com
1/4/2017

```
    // print the message (optional)
    // println(val);

    //if it's there, clear the buffer, and send a request for data
    if (firstContact == false) {
        myPort.clear();
        firstContact = true;
    }

    else { //if we've already established contact, keep getting and parsing data
      //println(val);


  //turn incoming string into an integer
    seatVal = Integer.parseInt(val);

      if (seatVal == 1) {
      file.play();

      delay (soundInt);
     //removed file.stop(); and am no longer getting the Node id error message. Keep this in mind incase there's a buffer problem or something

    }

    record_data(seatVal, myTime.getTime(), myTime.getDate());
    }
  }

}




void record_data(int seatVal, String time, String date){

  if (seatVal == 1){
    seatState = "up";
  } else if (seatVal == 0){
```

Microcontroller Controlled Radio Trigger

Alex Nathanson | www.alexnathanson.com | alex@alexnathanson.com
1/4/2017

```
   seatState = "down or off";
 }

 fileInfo = ("toiletdata_january.txt");

 String lines[] = loadStrings(fileInfo);
 output=createWriter(fileInfo);

   for(int i=0; i<lines.length ; i++)
   {
     output.println(lines[i]);
   }
   output.println(date + " " + time + " " + seatState);
   output.flush();
   output.close();
   println(date + " " + time + " " + seatState);
 }

 class AlarmTime {
   int d;
   int m;
   int y;
   int min;
   int h;
   int s;

   String date;
   String time;
   String day;
   String month;
   String year;
   String minute;
   String hour;
   String second;

   // contructor - initializes all values
   AlarmTime () {
     d = day();    // Values from 1 - 31
     m = month();  // Values from 1 - 12
     y = year();   // 2003, 2004, 2005, etc.
```

Microcontroller Controlled Radio Trigger

Alex Nathanson | www.alexnathanson.com | alex@alexnathanson.com
1/4/2017

```
    min = minute();  // Values from 0 - 59
    h = hour();    // Values from 0 - 23
    s = second();

    day = String.valueOf(d);
    month = String.valueOf(m);
    year = String.valueOf(y);
    date = (month + "/" + day + "/" + year + " ");

    minute = String.valueOf(min);
    hour = String.valueOf(h);
    second = String.valueOf(s);

    time = (hour + ":" + minute + ":" + second);
  }

  //method
  public String getTime() {

    min = minute();  // Values from 0 - 59
    h = hour();    // Values from 0 - 23
    s = second();

    minute = String.valueOf(min);
    hour = String.valueOf(h);
    second = String.valueOf(s);

    time = (hour + ":" + minute + ":" + second);
    //println(time);
    return time;
  }

  //method
  public String getDate(){
    d = day();    // Values from 1 - 31
    m = month();  // Values from 1 - 12
    y = year();   // 2003, 2004, 2005, etc.

    day = String.valueOf(d);
    month = String.valueOf(m);
    year = String.valueOf(y);
```

Microcontroller Controlled Radio Trigger

Alex Nathanson | www.alexnathanson.com | alex@alexnathanson.com
1/4/2017

```
    date = (month + "/" + day + "/" + year);

   // println(date);
    return date;
  }

}
```

Housing

To be added in the future.

References

Hardware and Software
https://processing.org/
https://www.arduino.cc/
https://learn.adafruit.com/introducing-pro-trinket/overview

Flash Radio Trigger Circuits and Hacks
http://www.glacialwanderer.com/hobbyrobotics/?p=10
http://www.satsleuth.com/Schematics.aspx?Category=Photo
http://www.diyphotography.net/diy-hot-shoe-adapter/
http://www.diyphotography.net/very_cool_optical_slave_unit/