Using IR-Sensors to Limit the Screen-On Time of a Computer

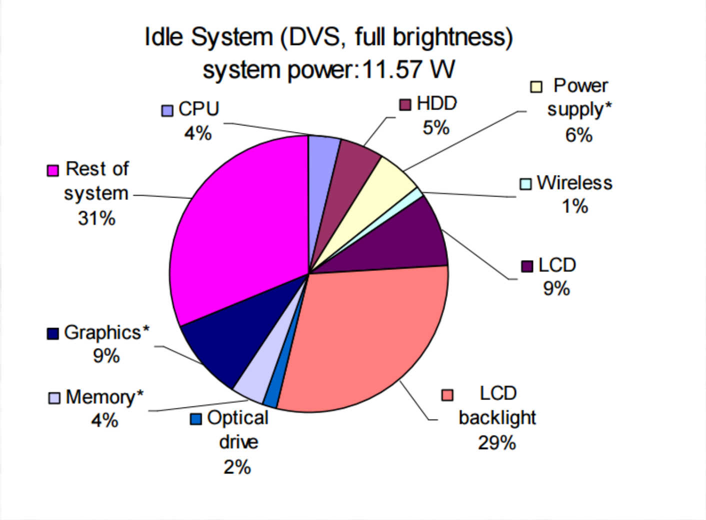
Akshath Jain
North Allegheny Senior High School
9th Grade

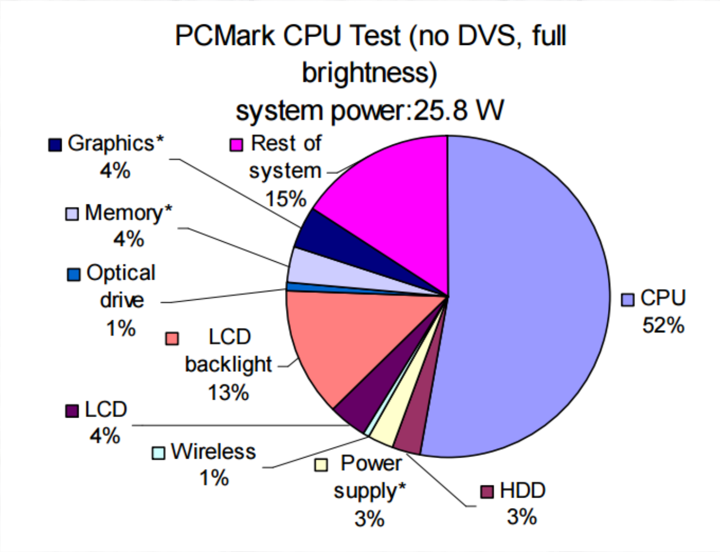
Problem

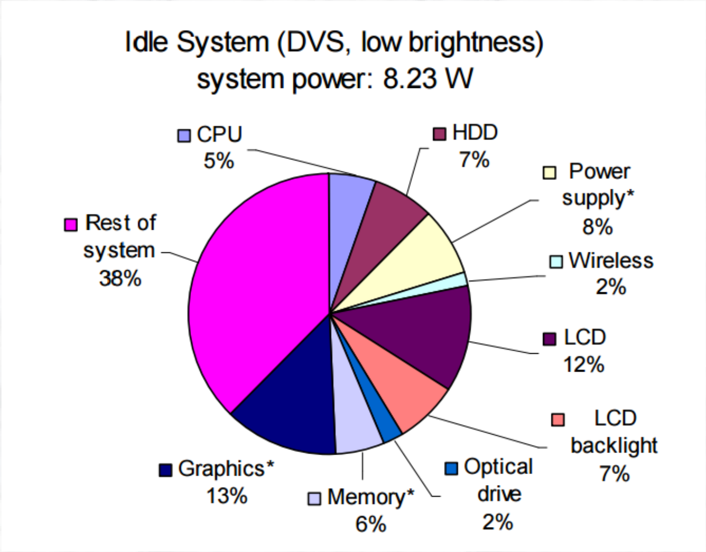
- Technological advancements make products faster
  - Require more power
- Ameliorations have been made in all parts of a computer
  - Except for one
- Betterments of the backbone of our technology has ebbed to a nonexistent level
- The Battery

Background Research

- Power consumption breakdown on a computer–
  1. CPU
  2. Screen
  3. GPU
  4. HDD/SSD

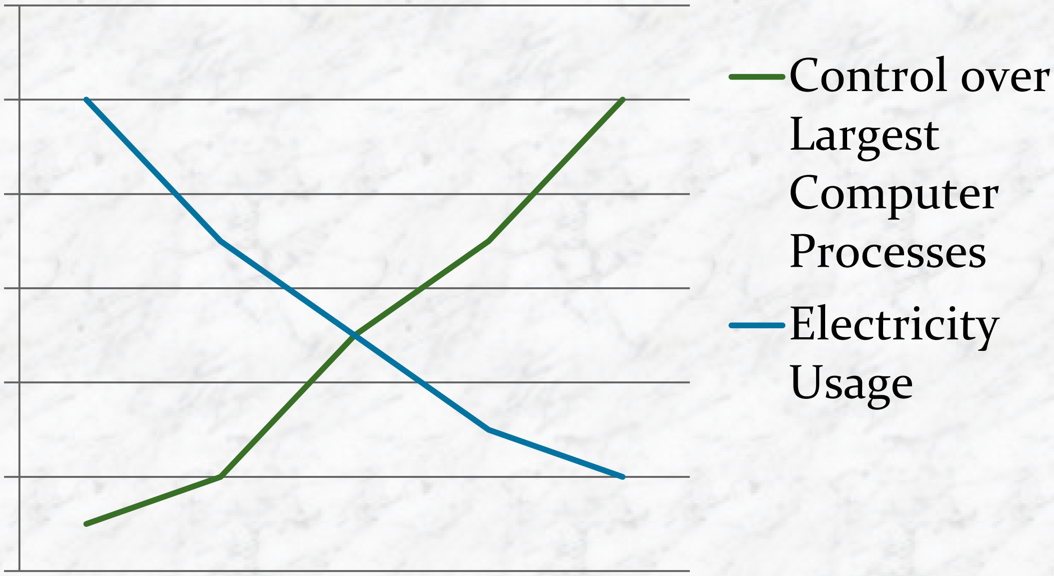






My Hypothesis

- If I can create a device that can control the largest energy consuming processes of a computer, then I will be able to mitigate the electricity requirements on that computer.



(This is to illustrate my hypothesis. Not shown to scale)

Purpose

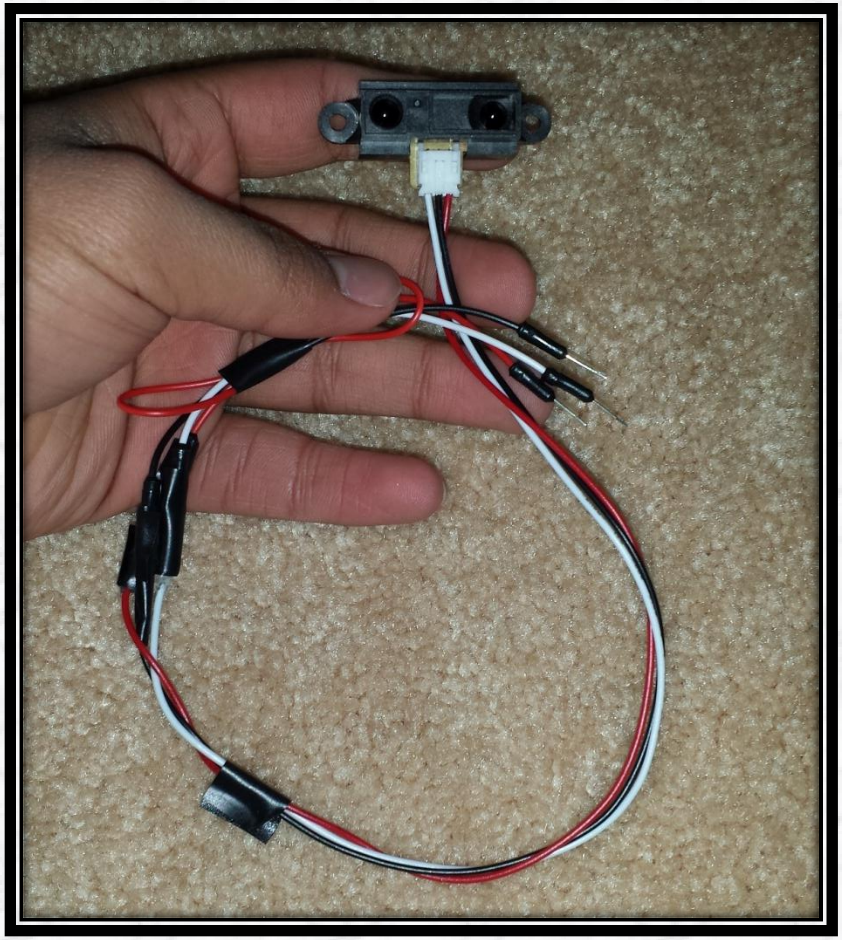
- Create a device that
  - Utilizes user tracking algorithms
  - To determine the user's status at the computer
  - Take the appropriate action based on the user's status
  - Is an alternative to current power saving techniques

Engineering & Design Process

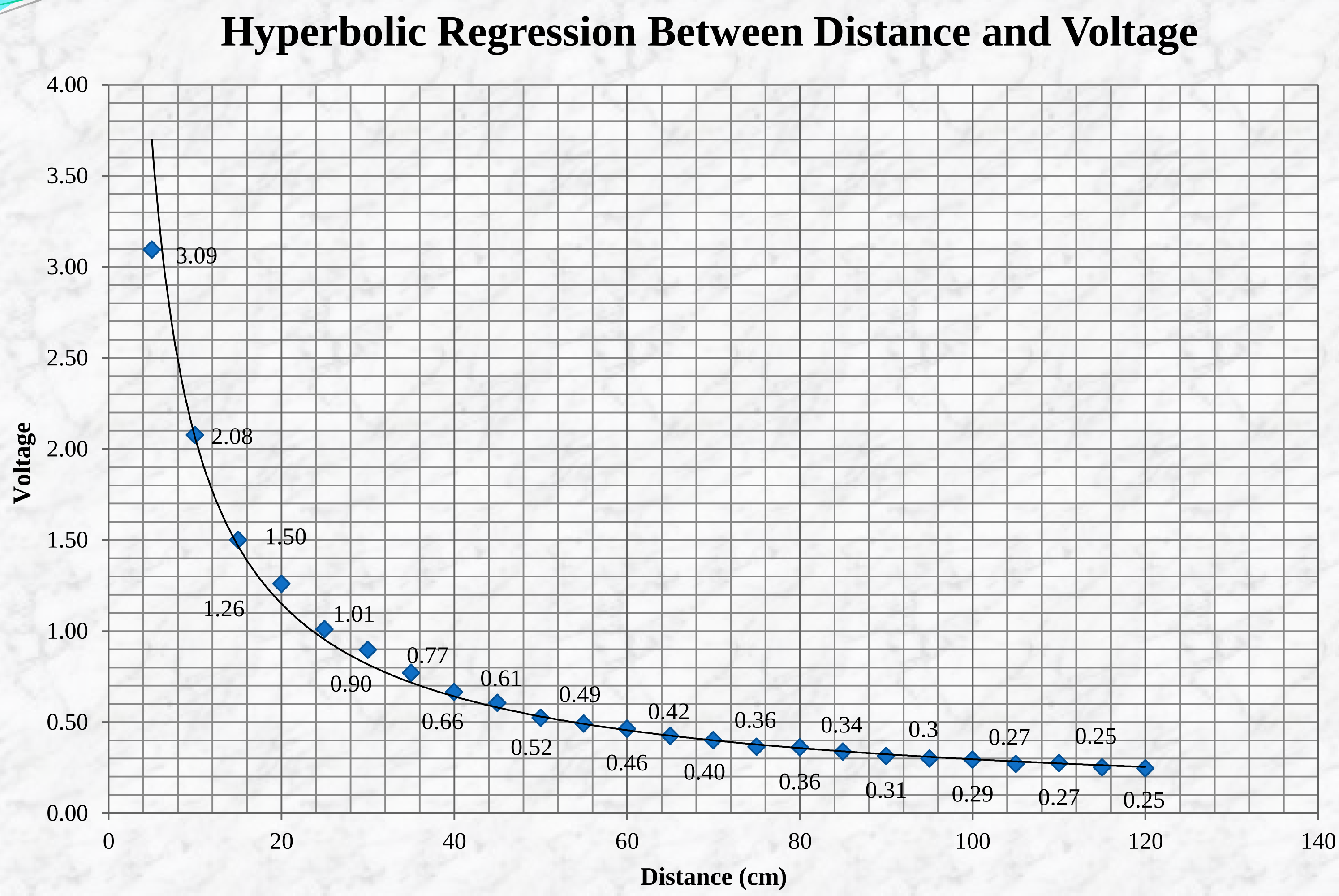
- Required parts:
  - A sensor
  - A microcontroller to control the sensor
  - A structure to house the various components

About the IR-Sensor

- Takes in 5 volts
- Integral combination of the:
  - Transmitter
    - IRED
  - Receiver
    - PSD
- Uses encapsulated triangulation techniques to gauge distance
- Utilizes analog output
  - High voltage – shorter distance
  - Low voltage – longer distances



IR Sensor Output



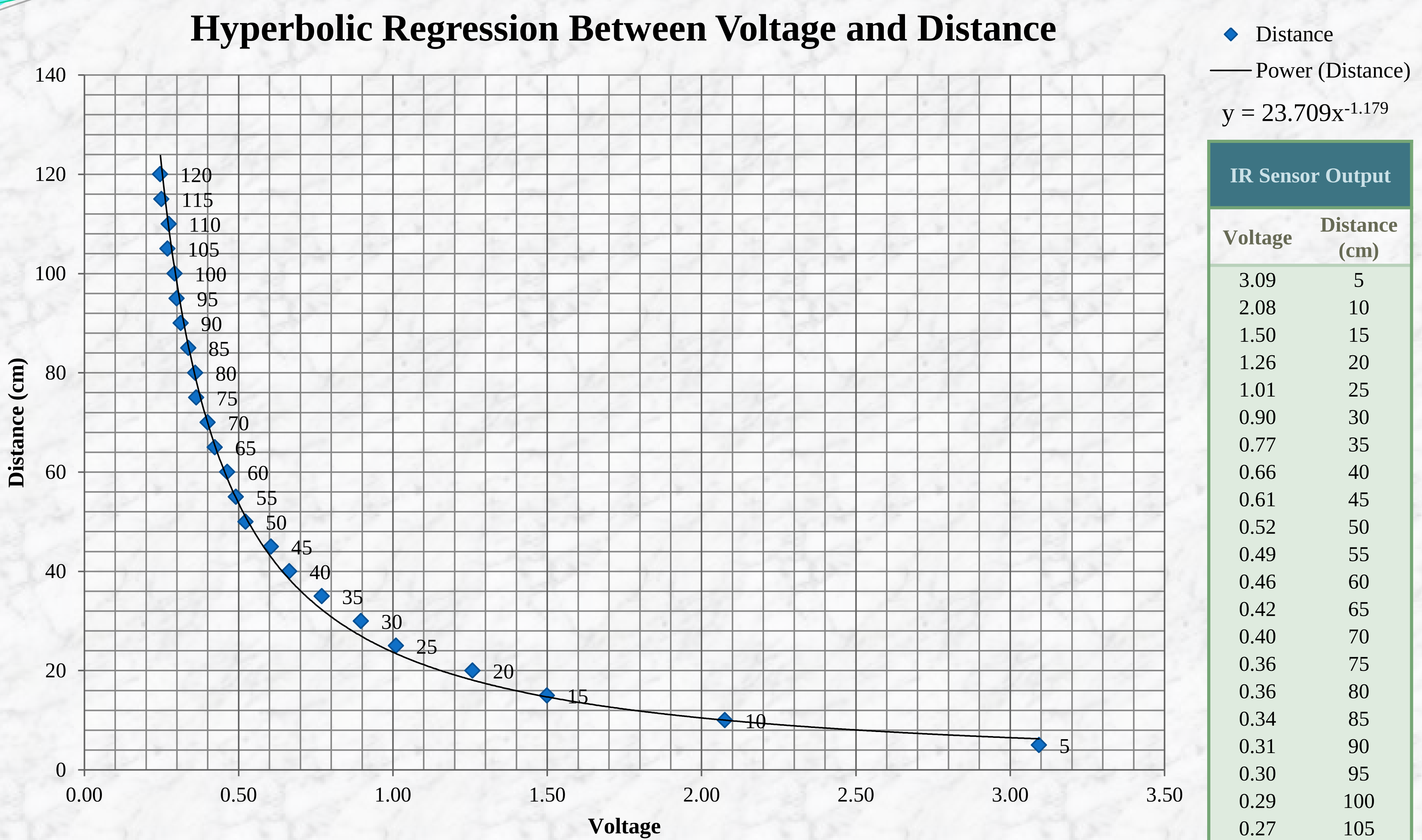
Voltage

Power (Voltage)

$y = 14.363x^{-0.843}$

IR Sensor Output	
Distance (cm)	Voltage
5	3.09
10	2.08
15	1.50
20	1.26
25	1.01
30	0.90
35	0.77
40	0.66
45	0.61
50	0.52
55	0.49
60	0.46
65	0.42
70	0.40
75	0.36
80	0.36
85	0.34
90	0.31
95	0.30
100	0.29
105	0.27
110	0.27
115	0.25
120	0.25

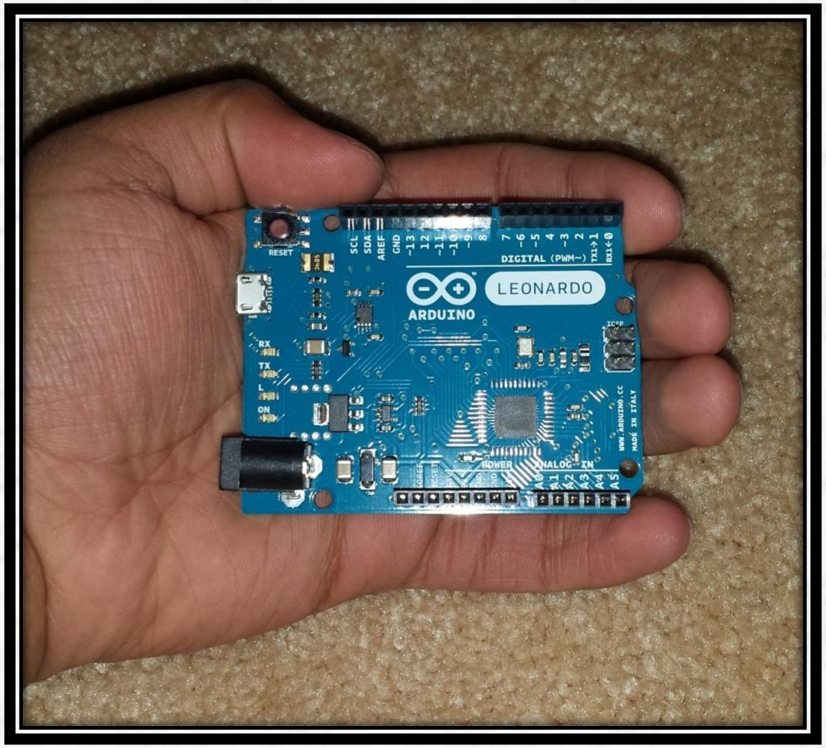
IR Sensor Output



IR Sensor Output	
Voltage	Distance (cm)
3.09	5
2.08	10
1.50	15
1.26	20
1.01	25
0.90	30
0.77	35
0.66	40
0.61	45
0.52	50
0.49	55
0.46	60
0.42	65
0.40	70
0.36	75
0.36	80
0.34	85
0.31	90
0.30	95
0.29	100
0.27	105
0.27	110
0.25	115
0.25	120

Arduino Leonardo

- Microcontroller
- The CPU is the ATmega32u4 chipset
- Emulates a USB 2.0 device
- Computer connection established via a micro USB cable
- Requires very little power
  - Only 8 mAh



About the Device

- I call it **I.R.E.S.**

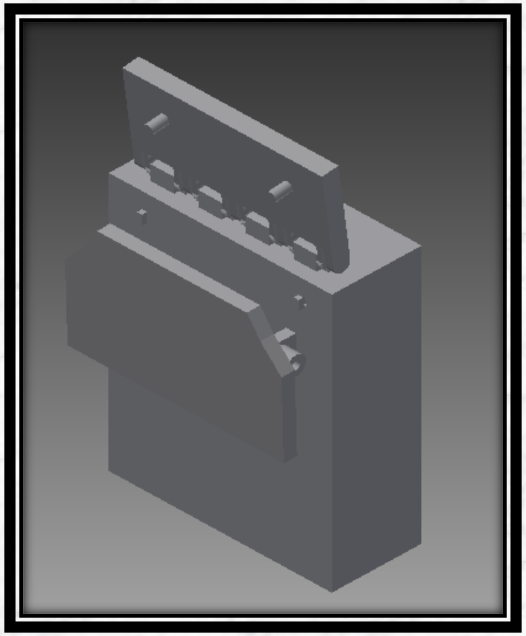
I – Infra-

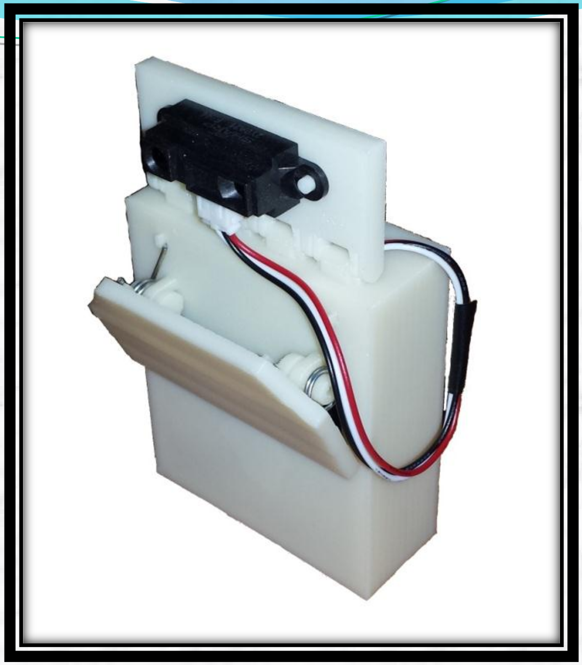
R – Red

E – Energy

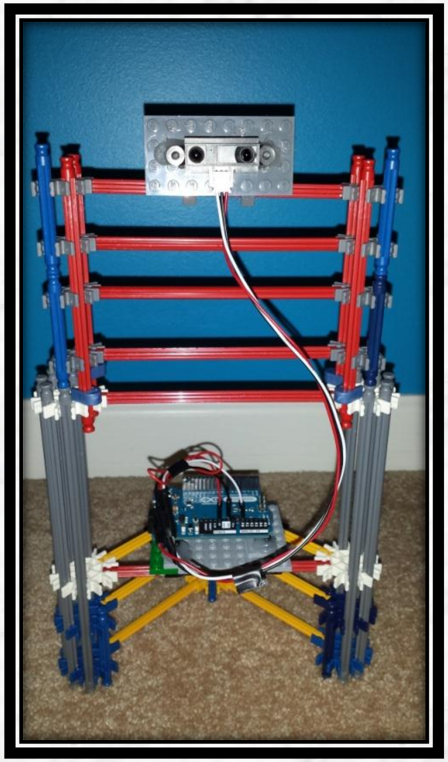
S – Saving

(Device)

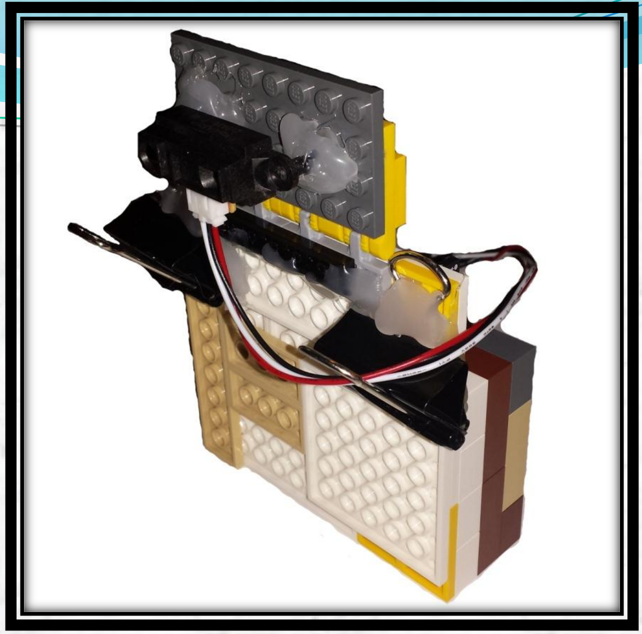




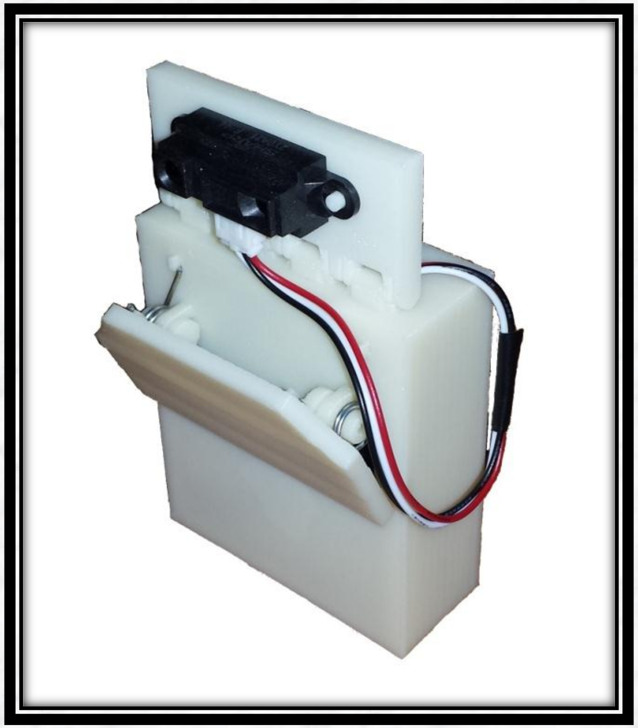
IRES Versions



IRES v1

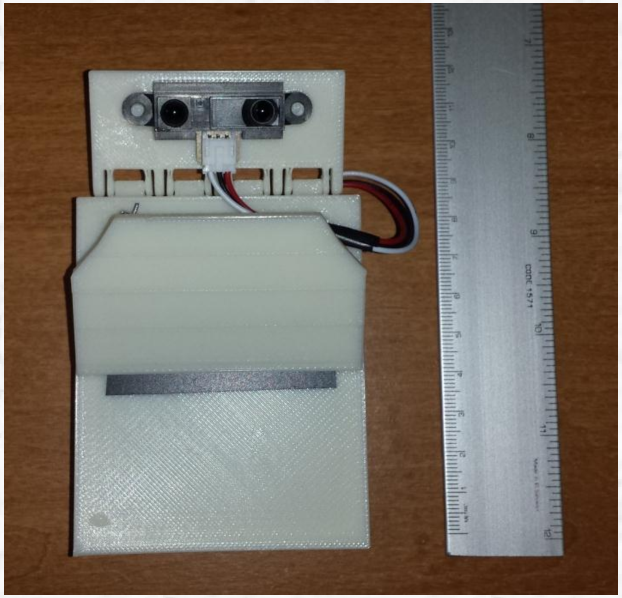


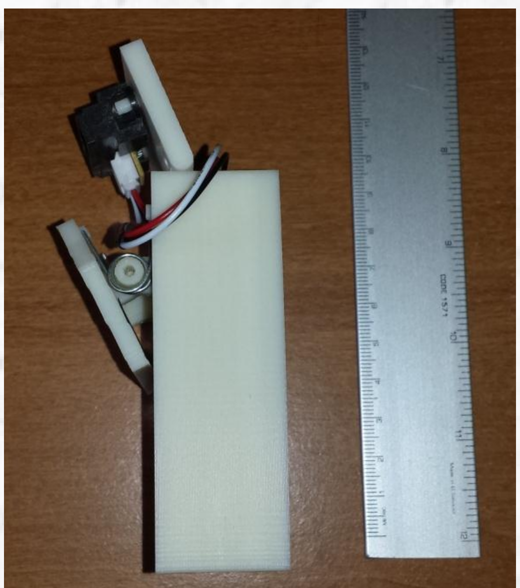
IRES v2

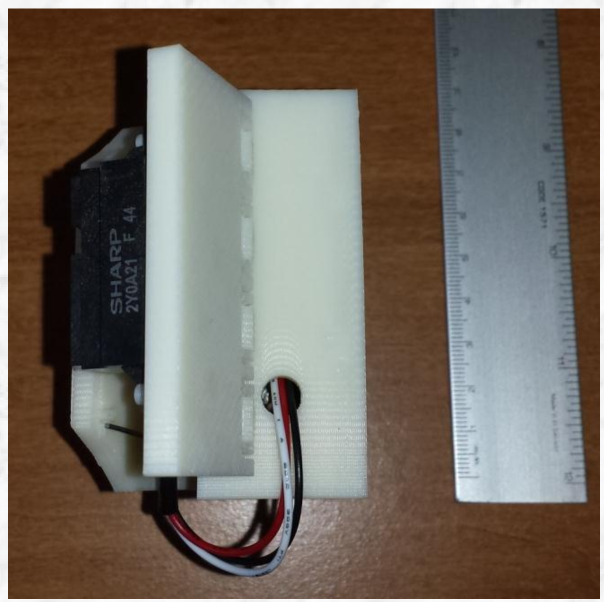


IRES v3

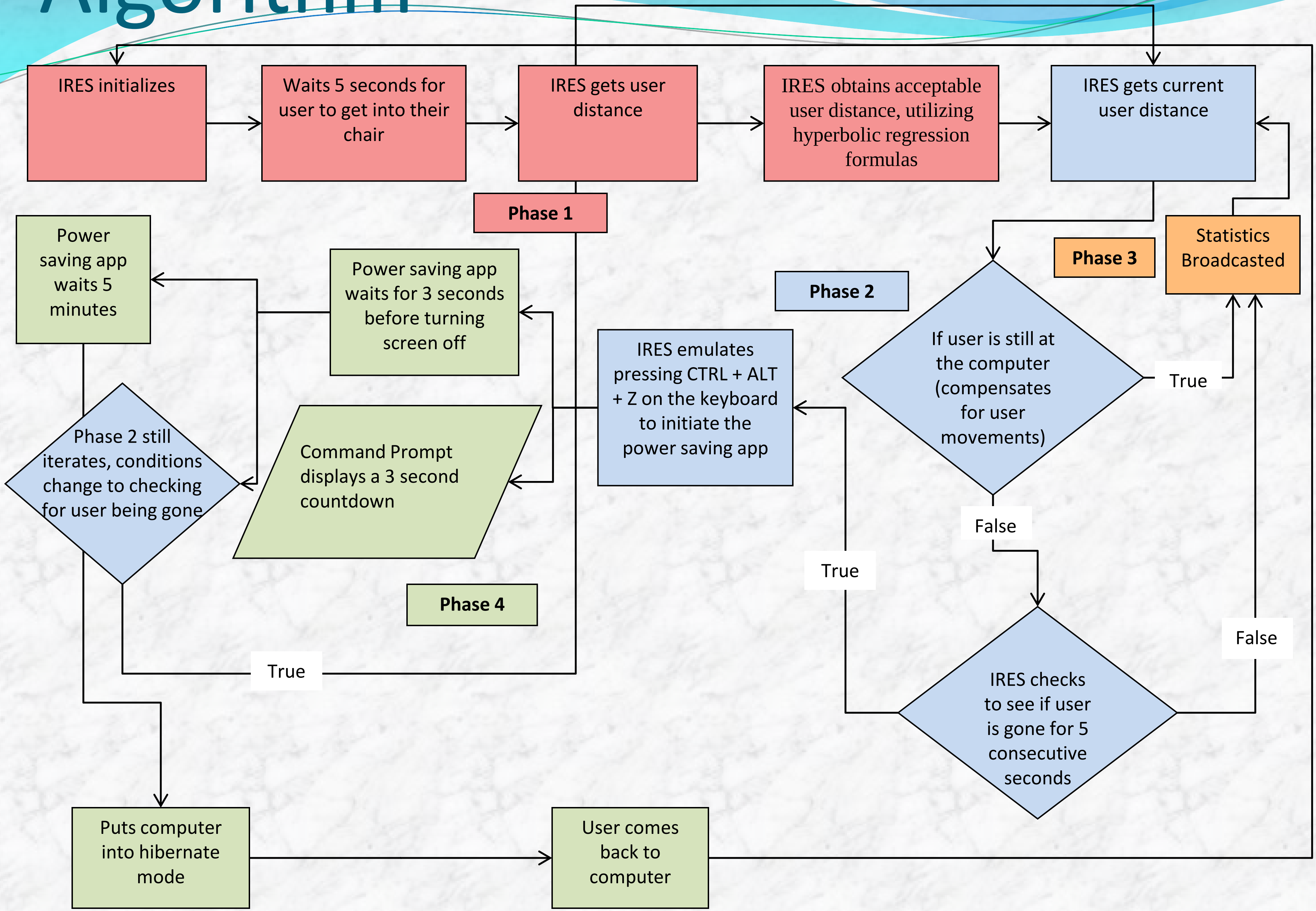
IRES Pictures







Algorithm


IRES initializes
Waits 5 seconds for user to get into their chair
IRES gets user distance
IRES obtains acceptable user distance, utilizing hyperbolic regression formulas
IRES gets current user distance
Phase 1
Power saving app waits 5 minutes
Power saving app waits for 3 seconds before turning screen off
Phase 2
Phase 3
Statistics Broadcasted
If user is still at the computer (compensates for user movements)
True
IRES emulates pressing CTRL + ALT + Z on the keyboard to initiate the power saving app
Phase 2 still iterates, conditions change to checking for user being gone
Command Prompt displays a 3 second countdown
False
True
Phase 4
False
IRES checks to see if user is gone for 5 consecutive seconds
True
Puts computer into hibernate mode
User comes back to computer

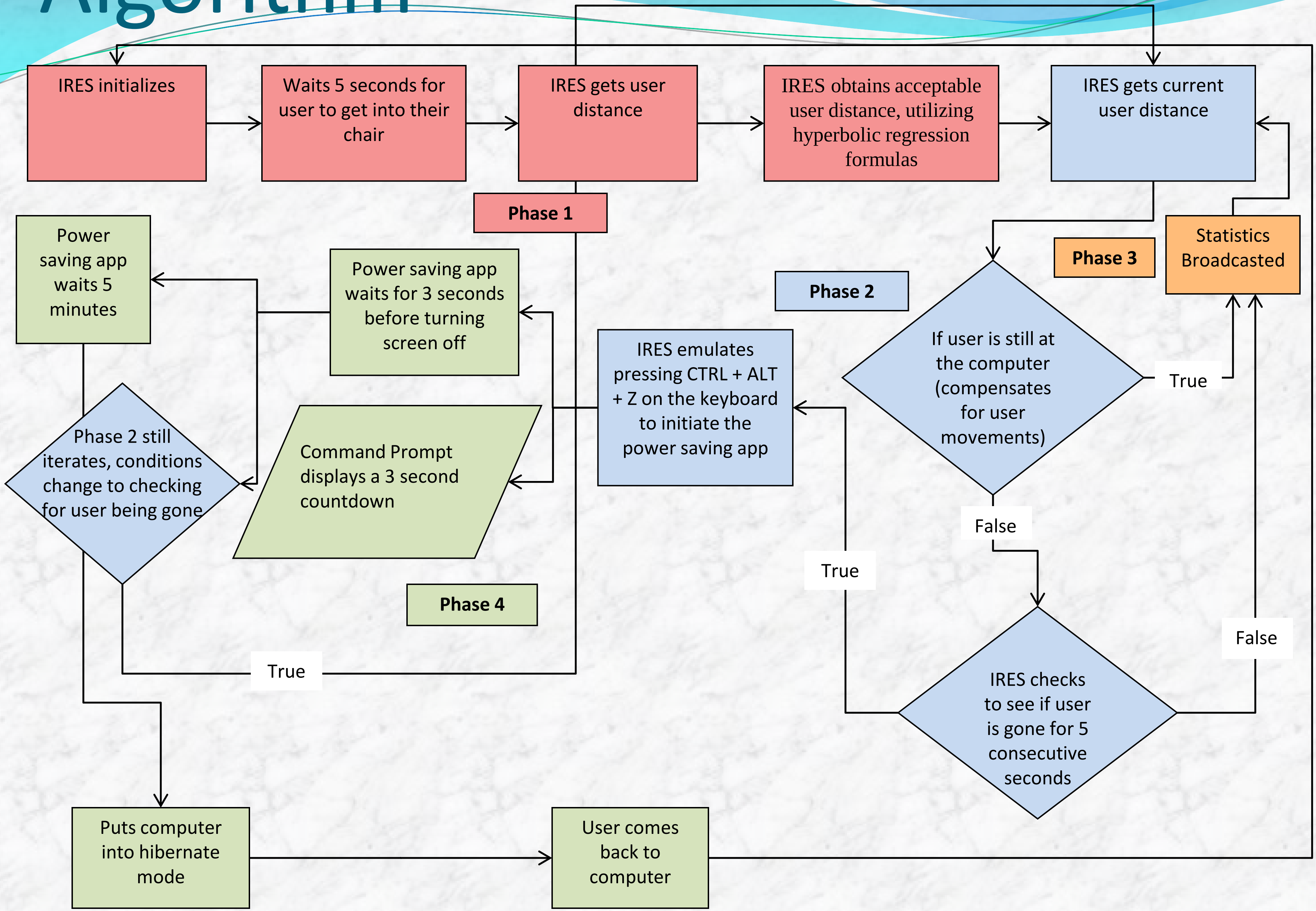
Arduino Code

```
/*
Name: Akshath Jain
 Date: 3/14/15
 Purpose: To create a program for IRES that is GUI compatible and efficient
 */

const int IRPIN = A0;
int time = 0, userStatus = 0; //for userStatus, 0 is false, 1 is true, and 2 is left the computer
double initialUserDistance, currentUserDistance, acceptableDistance;

void setup(){
  pinMode(IRPIN, INPUT); //Sets the ir sensor as an input
  Keyboard.begin();
  Serial.begin(9600);
  delay(5000); //Waits 5 second
  initialUserDistance = getDistance();
  getAcceptableDistance();
}

void loop(){
  currentUserDistance = getDistance(); //Gets current user distance
  userStatus = 0;
  if(currentUserDistance < acceptableDistance && verifyUserDistance()) {
    keyPress();

    while(currentUserDistance < acceptableDistance){
      currentUserDistance = getDistance();

      if(currentUserDistance >= acceptableDistance && verifyUserGone())
        break;

      time++;
      printStats();
    }
  }
  time++; //keeps track of time
  printStats();
}
```

```
boolean verifyUserDistance(){ //return true if user is gone for 5 consecutive seconds
  userStatus = 1;

  for(int i = 0; i < 5; i++){ //This loop tests to make sure that the user is gone, makes sure for 5 seconds
    currentUserDistance = getDistance(); //Gets user distance

    if(currentUserDistance >= acceptableDistance){ //Checks to see if user is at the computer
      userStatus = 0;
      return false; //If the user is still at the computer
    }

    time++;
    printStats();
  }
  userStatus = 2;
  return true;
}

boolean verifyUserGone(){
  userStatus = 1;

  for(int i = 0; i < 5; i++){
    currentUserDistance = getDistance(); //Gets user distance

    if(currentUserDistance < acceptableDistance){
      userStatus = 2;
      return false; //If user is still gone
    }

    time++;
    printStats();
  }
  userStatus = 0;
  return true;
}

double getDistance(){ //gets the user's distance from the ir sensor
  double temp = 0;

  for(int i = 0; i < 1000; i++){ //This loop will get a thousand readings
    temp += analogRead(IRPIN);
    delay(1);
  }
  return temp/1000; //And this will return the average of those thousand readings
}
```

```
double getAcceptableDistance(){
  double v = initialUserDistance * 5.0/1023, d;
  d = 23.709*pow(v, -1.179); //voltage to distance
  d += 20;//this allows the user to move back for 30 cm, equivalent to leaning back.
  acceptableDistance = 14.363*pow(d, -0.843) * 1023.0/5;   //distance to voltage
}

void keyPress(){ //This function initiates the keypresses required to turn the screen off
  char ctrlKey = KEY_LEFT_CTRL; //This is how the Arduino recognizes the control key
  char altKey = KEY_LEFT_ALT; //This is how the Arduino recognizes the alt key
  char zKey = 'z'; //This is how the Arduino recognizes the z key
  Keyboard.press(ctrlKey); //Arduino presses the control key
  delay(10); //Delays for 10 milliseconds
  Keyboard.press(altKey); //Arduino presses the alt key
  delay(10);
  Keyboard.press(zKey); //Arduino presses the z key
  delay(10);
  Keyboard.releaseAll(); //Arduino releases all the key's that are being pressed (control + alt + z)
}

void printStats(){
  Serial.print(initialUserDistance);
  Serial.print(' ');
  Serial.print(acceptableDistance);
  Serial.print(' ');
  Serial.print(currentUserDistance);
  Serial.print(' ');
  Serial.print(time);
  Serial.print(' ');
  Serial.print(userStatus);
  Serial.print(';');
}
```

Algorithm

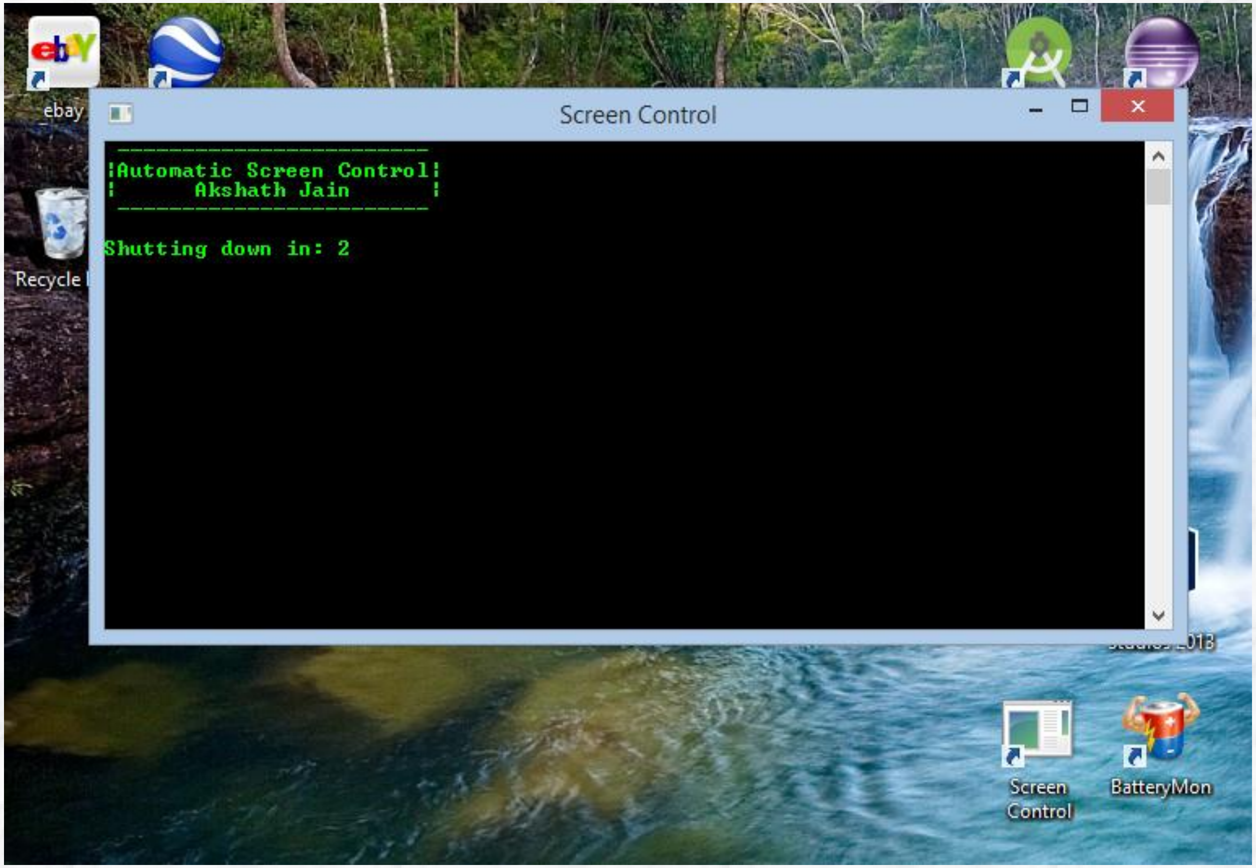


IRES initializes
Waits 5 seconds for user to get into their chair
IRES gets user distance
IRES obtains acceptable user distance, utilizing hyperbolic regression formulas
IRES gets current user distance
Phase 1
Power saving app waits 5 minutes
Power saving app waits for 3 seconds before turning screen off
Phase 3
Statistics Broadcasted
Phase 2
If user is still at the computer (compensates for user movements)
True
IRES emulates pressing CTRL + ALT + Z on the keyboard to initiate the power saving app
Phase 2 still iterates, conditions change to checking for user being gone
Command Prompt displays a 3 second countdown
False
True
Phase 4
False
IRES checks to see if user is gone for 5 consecutive seconds
True
Puts computer into hibernate mode
User comes back to computer

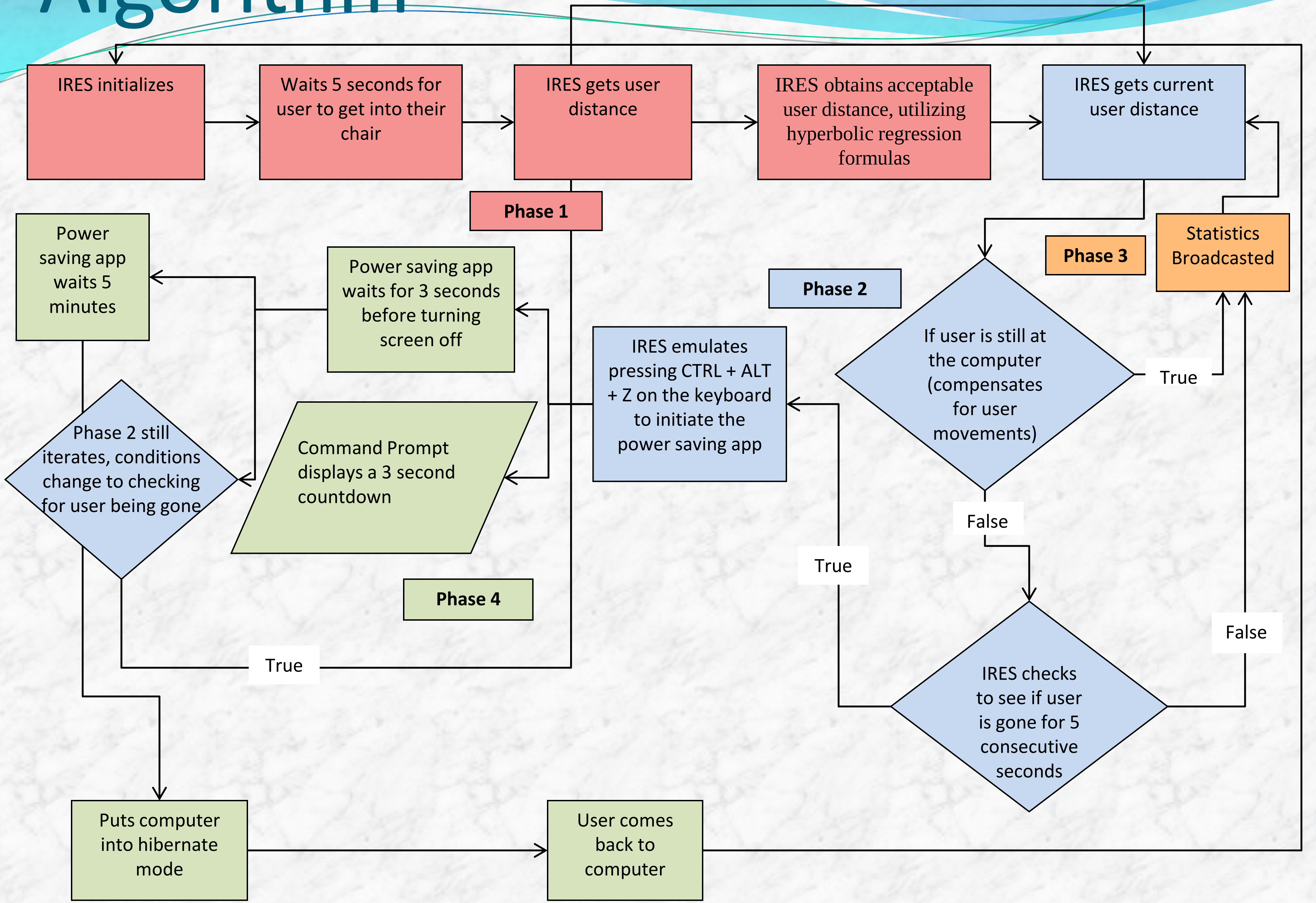
Application Code

```cpp
/*
Name: Akshath Jain
Date: 11/27/14
Purpose: To control my computer screen
*/

#include <iostream>
#include <Windows.h>
#include <PowrProf.h>
#include <time.h>
#pragma comment(lib, "PowrProf.lib")
using namespace std;

int main()
{
    system("color 0a");

    cout << " ------------------------" << endl;
    cout << "|Automatic Screen Control|" << endl;
    cout << "|      Akshath Jain      |" << endl;
    cout << " ------------------------" << endl;

    cout << "\nShutting down in: ";
    for (int i = 3; i >= 0; i--){
        cout << i << "\b"; //Backspace
        Sleep(1000); //Waits 1 seconds, total of 4 seconds
    }
    SendMessage(HWND_BROADCAST, WM_SYSCOMMAND, SC_MONITORPOWER, (LPARAM)2); //Turns screen off
    Sleep(300000); //waits 5 minutes
    SetSuspendState(true, true, true); //hibernates computer
}
```

Application



Algorithm


IRES initializes
Waits 5 seconds for user to get into their chair
IRES gets user distance
IRES obtains acceptable user distance, utilizing hyperbolic regression formulas
IRES gets current user distance
Phase 1
Power saving app waits 5 minutes
Power saving app waits for 3 seconds before turning screen off
Phase 3
Statistics Broadcasted
Phase 2
If user is still at the computer (compensates for user movements)
True
IRES emulates pressing CTRL + ALT + Z on the keyboard to initiate the power saving app
Phase 2 still iterates, conditions change to checking for user being gone
Command Prompt displays a 3 second countdown
False
True
Phase 4
False
IRES checks to see if user is gone for 5 consecutive seconds
True
Puts computer into hibernate mode
User comes back to computer

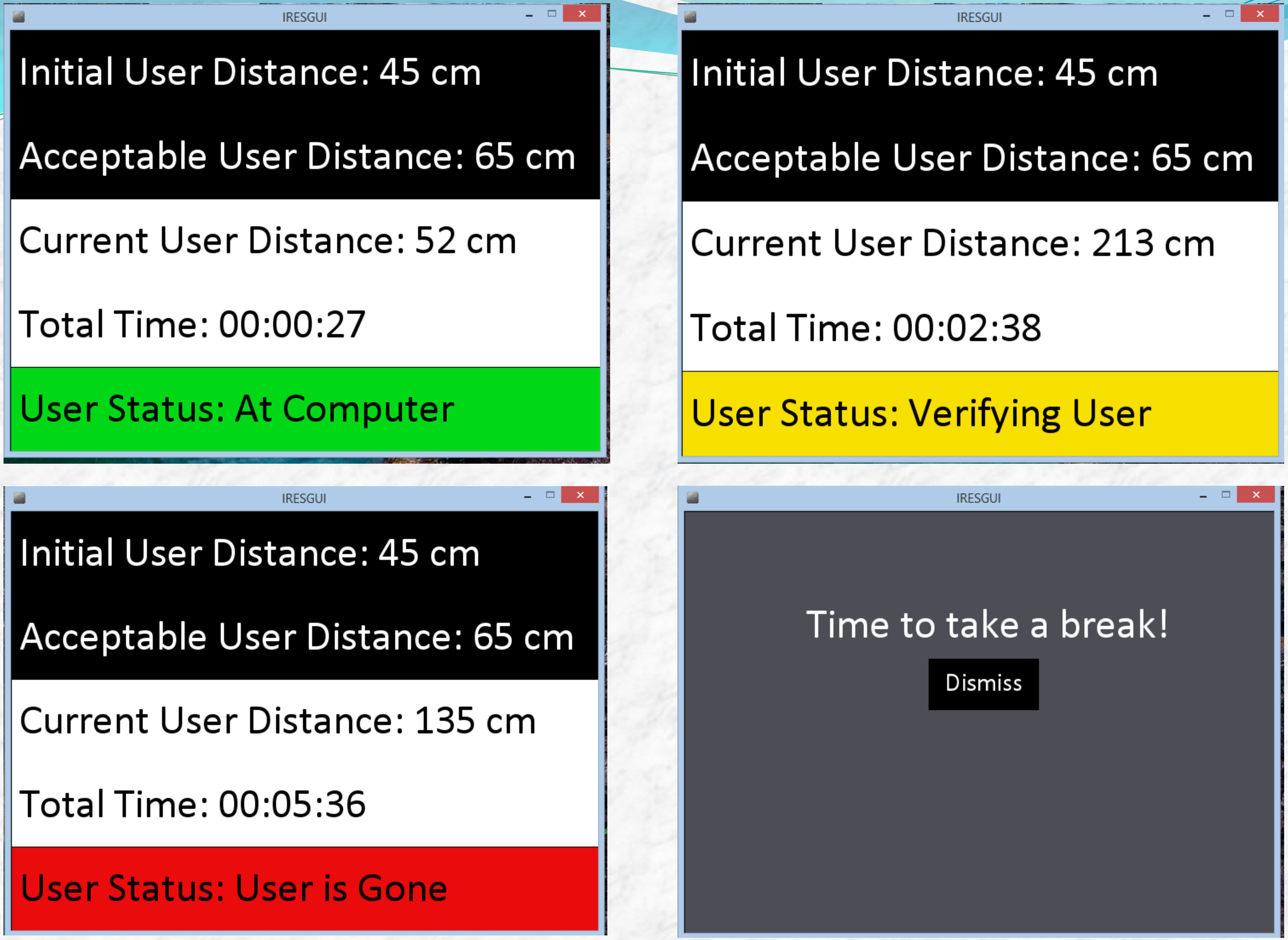
Graphical User Interface

```
/*
Name: Akshath Jain
 Date: 3/14/15
 Purpose: Creating a GUI for IRES
 */
import processing.serial.*;

Serial port;
PFont font;
int size = 5;
String IRESData[] = {"00", "00", "00", "0", " "};
String headings[] = {"Initial User Distance: ", "Acceptable User Distance: ", "Current User Distance: ", "Total Time: ", "User Status: "};
String subHeadings[] = {" cm", " cm", " cm", "", ""};
int userStatusColor = #ffffff, totalTimeInSeconds, currentTimeInSeconds = 120000;

void setup(){
  size(700, 500);
  port = new Serial(this, "COM3", 9600);
  port.bufferUntil(';'); //specifies how often serialEvent() will iterate
  font = loadFont("Calibri-50.vlw");
  textFont(font);
}

void draw(){
  if(totalTimeInSeconds > currentTimeInSeconds){
    if((mouseX > 290 && mouseX < 420) && (mouseY > 175 && mouseY < 235)){ //if mouse is over the dismiss button
      showMessage(#67676F, #F5F5F5); //calls function show message
      if(mousePressed == true){ //if mouse is pressed
        currentTimeInSeconds = totalTimeInSeconds + 10000; //reminds the user to take a break
      }                                                    //once every 10 minutes after 2 hours
    }
    else
      showMessage(#000000, #ffffff);
  }
  else{
    setBackground();
    for (int i = 0; i < size; i++){
      if (i < 2) //makes text white
        fill(255, 255, 255);
      else //makes this text black
        fill(0, 0, 0);
      text(headings[i] + IRESData[i] + subHeadings[i], 10, (65 + i * 100));
    }
  }
}
```

```
void serialEvent(Serial port){
  char endStop[] = {' ', ' ', ' ', ' ', ';'};
  for (int i = 0; i < size; i++){
    IRESData[i] = port.readStringUntil(endStop[i]);
    IRESData[i] = IRESData[i].substring(0, IRESData[i].length() - 1);
  }
  fixDistance();
  fixTime();
  fixUserStatus();
}

void setBackground(){
  background(0, 0, 0); //sets the bacground to black
  fill(255, 255, 255); //makes a white rectangle (x,y,width,height)
  rect(0, 200, 700, 200);

  fill(userStatusColor); //makes a colored rectangle depending on where the user is
  rect(0, 400, 700, 100);
}

void showMessage(int buttonColor, int textColor){
  background(76,77,85);

  fill(76,77,85);
  rect(0,0,700,500);

  fill(255,255,255);
  text("Time to take a break!", 145, 150);

  fill(buttonColor);
  rect(290, 175, 130, 60); //x,y,length,width

  fill(textColor);
  textSize(30);
  text("Dismiss", 310, 213);

  textSize(50);
}
```

```
void fixDistance(){
  double v, d;
  for (int i = 0; i < 3; i++){
    v = Double.parseDouble(IRESData[i]) * 5.0/1023; //converts the Arduino's [0,1023] into voltage from [0,5]
    d = 23.709*pow((float)v, -1.179); //voltage to distance in cm
    IRESData[i] = Integer.toString((int)(d + .5)); //puts data back into array IRESData and
  }                                                //rounds distance, converts to integer, and turns it to a string
}

void fixTime(){
  int h, m, s;

  s = Integer.parseInt(IRESData[3]);
  totalTimeInSeconds = s; //total time in seconds, stored to check if user has been working too long

  m = s / 60; //converts total seconds into the total number of whole minutes
  s %= 60; //converts total seconds into minutes and seconds, disregarding minutes because this is modular division
  h = m/60; //converts the total number of minutes into total number of whole hours
  m %= 60; //converts total minutes into minutes, it disregards hours because this is modular division
```

```
  int t[] = {h, m, s};
  String time[] = {"", "", ""};

  for (int i = 0; i < 3; i++){
    if (t[i] >= 0 && t[i] <= 9)
      time[i] = '0' + Integer.toString(t[i]); //converts an int to a string and adds a 0 in front of it
    else
      time[i] = Integer.toString(t[i]); //converts an int to a string
  }
  IRESData[3] = time[0] + ':' + time[1] + ':' + time[2]; //puts fixed data back into array IRESData at index 3
}

void fixUserStatus(){
  int status = Integer.parseInt(IRESData[4]);
  switch(status){
  case 0:
    userStatusColor = #00D817;
    IRESData[4] = "At Computer";
    break;
  case 1:
    userStatusColor = #F7DF00;
    IRESData[4] = "Verifying User";
    break;
  case 2:
    userStatusColor = #EA0C0C;
    IRESData[4] = "User is Gone";
    break;
  }
}
```


IRESGUI
Initial User Distance: 45 cm
Acceptable User Distance: 65 cm
Current User Distance: 52 cm
Total Time: 00:00:27
User Status: At Computer
IRESGUI
Initial User Distance: 45 cm
Acceptable User Distance: 65 cm
Current User Distance: 213 cm
Total Time: 00:02:38
User Status: Verifying User
IRESGUI
Initial User Distance: 45 cm
Acceptable User Distance: 65 cm
Current User Distance: 135 cm
Total Time: 00:05:36
User Status: User is Gone
IRESGUI
Time to take a break!
Dismiss

Testing IRES

- Testing IRES, a.k.a. watching paint dry
- Control
  - 3 Trials
- 6 Test Cases
  - 3 Trials
- Surveys from 15 volunteers
  - Help form test cases
- 60 minutes each
- Tested using BatteryMon, by PassMark Software

The Test Cases

<u>Control</u> – Running the computer for 60 minutes without any power saving techniques

1. Power consumption of IRES (the device itself)
2. Efficiency of the computer's default power saving techniques.
3. Efficiency of IRES

The Test Cases (cont.)

4. Laptop efficiency with actual usage
5. IRES efficiency with actual usage
6. Desktop efficiency with actual usage

Questionnaire

Proctor’s Name:
Participant’s Name:
Date:

Computer Usage

---

1. Do you use a computer daily?
2. How long do you use it for on average?
3. Do you primarily use a desktop or a laptop?
   a. (If laptop) Is your laptop plugged in and charging while you work?
      i. (If yes) Why do you keep your laptop plugged in? Is it because of a short battery life?
         1. (If yes) Would you benefit if your battery life was better, giving you more versatility in places you are able to work?

Power Saving Techniques

---

4. Are you aware of any power-saving techniques used by your computer (i.e. screen turns off, computer goes to sleep, etc.)?
   a. (If Yes) How long until the screen times out? (With absolute certainty)
   b. How long until your computer goes to sleep? (With absolute certainty)

Work Habits

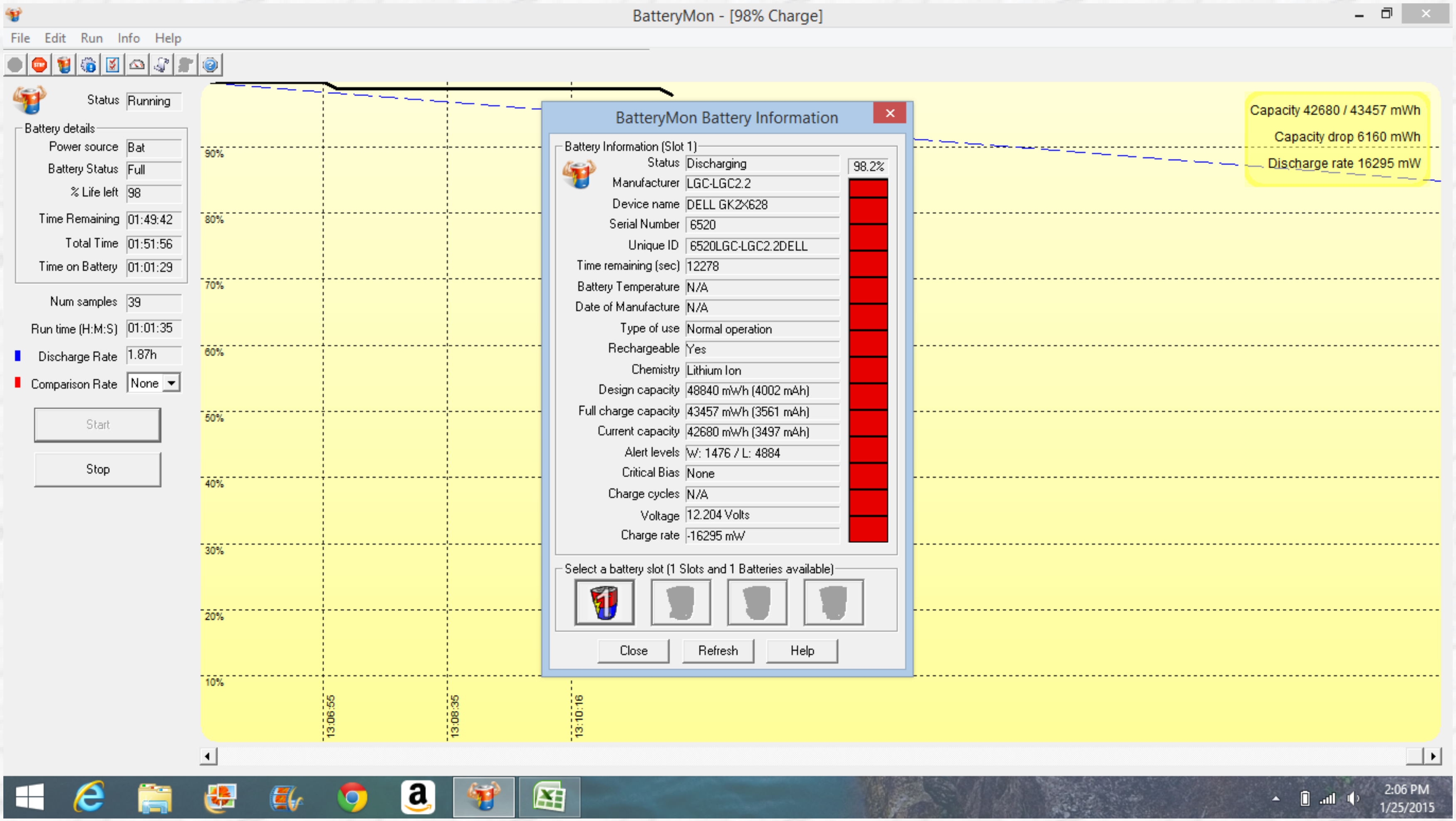
---

5. When you work, do you ever take short breaks away from your computer? This can include anytime that you are away from your computer.
   a. (If yes) For how long do you work at a time, and then take a break?
   b. What is the duration of your break?

Questionnaire Results

Questionaire Results												
	Computer Usage						Power Saving Techniques			Computer Habits		
	1	2	3	3a	3ai	3ai1	4	4a	4b	5	5a	5b
	Y/N	Hours	Laptop (L)/ Desktop (D)	Y/N	Convenience (C)/ Short Battery Life (SBL)/ Other (O)	Y/N	Y/N	Minutes	Minutes	Y/N	Hours	Minutes
Person												
1	Yes	10	L	Yes	C	Yes	Yes	10	15	Yes	2	10
2	Yes	10	L	Yes	SBL	Yes	Yes	5	10	Yes	1	45
3	Yes	10	L	Yes	C	Yes	No	5	15	Yes	2	10
4	Yes	10	L	Yes	C	Yes	Yes	15	30	Yes	1	5
5	Yes	10	D				No	0	0	Yes	3	30
6	Yes	10	L	Yes	C	Yes	Yes	5	15	Yes	1	10
7	Yes	10	L	Yes	C	Yes	Yes	5	10	Yes	0.5	5
8	Yes	8	L	Yes	C	Yes	Yes	5	15	Yes	2	5
9	Yes	10	L	Yes	C	Yes	Yes	10	15	Yes	4	5
10	Yes	8	L	Yes	C	Yes	Yes	5	15	Yes	1	32.5
11	No	1/2	D				No	0	0	No		
12	Yes	8	L	Yes	SBL	Yes	No	5	15	Yes	1.75	12.5
13	Yes	8	L	Yes	C	Yes	Yes	5	15	Yes	1	10
14	Yes	8	D				Yes	0	0	Yes	2.50	12.5
15	Yes	8	L	Yes	C	Yes	Yes	5	15	Yes	1.25	7.5
Average		9.14						5.33	12.33		1.71	14.29

BatteryMon


BatteryMon - [98% Charge]
File Edit Run Info Help
Status Running
Battery details
Power source Bat
Battery Status Full
% Life left 98
Time Remaining 01:49:42
Total Time 01:51:56
Time on Battery 01:01:29
Num samples 39
Run time (H:M:S) 01:01:35
Discharge Rate 1.87h
Comparison Rate None
Start
Stop
Capacity 42680 / 43457 mWh
Capacity drop 6160 mWh
Discharge rate 16295 mW
BatteryMon Battery Information
Battery Information (Slot 1)
Status Discharging
98.2%
Manufacturer LGC-LGC2.2
Device name DELL GK2X628
Serial Number 6520
Unique ID 6520LGC-LGC2.2DELL
Time remaining (sec) 12278
Battery Temperature N/A
Date of Manufacture N/A
Type of use Normal operation
Rechargeable Yes
Chemistry Lithium Ion
Design capacity 48840 mWh (4002 mAh)
Full charge capacity 43457 mWh (3561 mAh)
Current capacity 42680 mWh (3497 mAh)
Alert levels W: 1476 / L: 4884
Critical Bias None
Charge cycles N/A
Voltage 12.204 Volts
Charge rate -16295 mW
Select a battery slot (1 Slots and 1 Batteries available)
Close
Refresh
Help
90%
80%
70%
60%
50%
40%
30%
20%
10%
13:06:55
13:08:35
13:10:16
2:06 PM
1/25/2015

The Results

IRES Test Results												
Tests	Start			End			Time Elapsed	Variables				
	Starting Percentage	Starting Battery Level		Ending Percentage	Ending Battery Level			Applications Open			IR-Sensor	
	%	mAh	mWh	%	mAh	mWh	min	Microsoft Excel	BatteryMon	Microsoft Word	Enabled	
Control												
1	100.00%	3792	43457	75.60%	2868	32867	60	X	X			
2	100.00%	3770	43457	76.00%	2866	33034	60	X	X			
3	100.00%	3770	43457	75.60%	2851	32856	60	X	X			
Average	**100.00%**	**3777**	**43,457.00**	**75.73%**	**2862**	**32,919**	**60**					
Test Case 1												
1	100.00%	3783	43457	75.10%	2840	32623	60	X	X		X	
2	100.00%	3778	43457	75.20%	2840	32667	60	X	X		X	
3	100.00%	3615	41614	75.40%	2725	31369	60	X	X		X	
Average	**100.00%**	**3725**	**42,842.67**	**75.23%**	**2802**	**32,220**	**60**					
Test Case 2												
1	100.00%	3590	43457	93.80%	3368	40770	60	X	X			
2	100.00%	3611	43457	91.70%	3313	39871	60	X	X			
3	100.00%	3665	43457	87.30%	3119	37929	60	X	X			
Average	**100.00%**	**3622**	**43,457.00**	**90.93%**	**3266.67**	**39,523.33**	**60**					
Test Case 3												
1	100.00%	3573	43457	97.80%	3492	42480	60	X	X		X	
2	100.00%	3561	43457	98.10%	3493	42624	60	X	X		X	
3	100.00%	3561	43457	98.20%	3497	42680	60	X	X		X	
Average	**100.00%**	**3565**	**43,457.00**	**98.03%**	**3494**	**42,594.67**	**60**					
Test Case 4												
1	100.00%	3788	43457	73.90%	2800	32123	60	X	X	X		
2	100.00%	3802	43457	73.30%	2788	31168	60	X	X	X		
3	100.00%	3629	41614	72.80%	2641	30292	60	X	X	X		
Average	**100.00%**	**3739.67**	**42,842.67**	**73.33%**	**2743.00**	**31,194.33**	**60**					
Test Case 5												
1	100.00%	3696	43457	84.50%	3122	36708	60	X	X	X	X	
2	100.00%	3692	43457	84.50%	3119	36719	60	X	X	X	X	
3	100.00%	3517	41614	85.50%	3007	35587	60	X	X	X	X	
Average	**100.00%**	**3635.00**	**42,842.67**	**84.83%**	**3082.67**	**36,338.00**	**60**					
Test Case 6												
1	100.00%	3818	43457	71.90%	2744	31224	60	X	X	X		
2	100.00%	3793	43457	73.60%	2793	32001	60	X	X	X		
3	100.00%	3652	41614	71.20%	2600	29626	60	X	X	X		
Average	**100.00%**	**3754.33**	**42842.67**	**72.23%**	**2712.33**	**30950.33**	**60**					

User Tracking Algorithm Testing

- 15 volunteers to test IRES
- 4 different test cases
  1. Pushing chair back and getting up
  2. Turning to the right
  3. Turning to the left
  4. Random – However the volunteer gets out of his/her chair
- 3 trials
- IRES registered user movements 100% of the time
  - (0 false positives)

Compiled Data

Overall Statistics				
12:3 Laptops to Desktops	Status Qou	IRES	Savings	IRES Energy Reduction
	ΔmAh	ΔmAh	mAh	%
Desktops	915.67	71.00	844.67	92.25%
Laptops	355.33	71.00	284.33	80.02%
Average w/ 12:3	**467.40**	**71.00**	**396.40**	**84.81%**
Desktops w/ Usage	1034.00	552.33	481.67	46.58%
Laptops w/ Usage	988.67	552.33	436.33	44.13%
Average w/ 12:3	**997.73**	**552.33**	**445.40**	**44.64%**

Savings Statistics								
Energy Statistics							Savings	
	# of Computers	Energy Consumption	Computer Usage	Energy Cost	Work Days	CO_2	IRES Optimal	IRES w/ Usage
	Computers (million)	kW (w/ 12:3)	Hours	¢ / kWh	Days	lbs	%	
Yes	289.89	0.12	9.14	12.46	261	1.855	84.81%	44.64%
No	20.71	0.12	0.5	12.46	121	1.855	84.81%	44.64%

Impacts

- Cost Effective
  - Approximately **$10,355,900,180 / year** on computers
  - **44.64%** mitigation
  - Equates to **$4,624,306,919 / year** savings
- Environment
  - Currently emmit **76,968,868.41 Tons** of CO2 / year on computers
  - Prevent **34,472,699.91 Tons** of CO2 from entering the atmosphere yearly
  - Automobiles in the US account for **1,522,000,000 Tons** of CO2 every year

Conclusion

- My hypothesis was correct, by creating a device that controlled the largest processes on a computer, I was able to mitigate power consumption.
- Save **$4,624,306,919** worth of electricity
- Prevent **34,472,699.91 Tons** worth of CO_2 entering the atmosphere

Practicality

- IR Sensor: $10.00
- Arduino: $20.00
- Cable: $2.00
- **Total: $32.00**

Acknowledgements

- Mr. Bruce Allen
- Mr. Keith Banks – NAI Technology Education Department
- Judges & Event coordinators

Bibliography

"The Absolute Beginner's Guide to Arduino." *Forefront.io*. Forefront.io, 1 Jan. 2013. Web. 13 Nov. 2014. <http://www.forefront.io/a/beginners-guide-to-arduino>.

"Create Keyboard Shortcuts to Open Programs." *Windows*. Microsoft, 2014. Web. 11 Nov. 2014. <http://windows.microsoft.com/en-us/windows/create-keyboard-shortcuts-open-programs#1TC=windows-7>.

"Getting Started with Arduino." *Arduino*. Arduino, n.d. Web. 11 Nov. 2014. <http://arduino.cc/en/Guide/HomePage>.

"How Does Infrared Work." *Wise Geek*. Conjecture, n.d. Web. 11 Nov. 2014. <http://www.wisegeek.org/how-does-infrared-work.htm#comments>.

Bibliography

"How much carbon dioxide is produced by burning gasoline and diesel fuel?" *U.S. Energy Information Administration*. US Department of Energy, 2015. Web. 3 Feb. 2015. <http://www.eia.gov/tools/faqs/faq.cfm?id=307&t=10>.

"How much carbon dioxide is produced per kilowatthour when generating electricity with fossil fuels?" *U.S. Energy Information Administration*. US Department of Energy, 2015. Web. 3 Feb. 2015. <http://www.eia.gov/tools/faqs/faq.cfm?id=74&t=11>.

"How much power does a computer use? And how much CO2 does that represent?" *Energuide*. Energuide, 2015. Web. 2 Feb. 2015. <http://www.energuide.be/en/questions-answers/how-much-power-does-a-computer-use-and-how-much-co2-does-that-represent/54>.

"Keyboard and Mouse Control." *Arduino*. Arduino, 2014. Web. 8 Dec. 2014. <http://arduino.cc/en/Tutorial/KeyboardAndMouseControl>.

Bibliography

Lyadvinsky, Kirill V. "Problem with SuspenState." *StackOverFlow*. Stack Exchange, 2015. Web. 17 Jan. 2015. <http://stackoverflow.com/questions/3303243/problem-with-setsuspendstate>.

Mahesri, Aqeel, and Vibhore Vardhan. "Power Consumption Breakdown on a Modern Laptop." *Springer Link*. Springer International, 2015. Web. 23 Jan. 2015. <http://link.springer.com/chapter/10.1007%2F11574859_12>.

"Monitor Laptop and UPS Batteries." *PassMark BatteryMon*. PassMark Software, 2015. Web. 5 Jan. 2015. <http://www.passmark.com/products/batmon.htm>.

O'Reilly, Dennis. "Calculate Your PC's Energy Use." *CNET*. CBS Interactive, 2014. Web. 11 Nov. 2014. <http://www.cnet.com/news/quirky-ge-unveil-new-affordable-smart-home-line/>.

"Recent Data." *U.S. Energy Information Administration*. U.S. Department of Energy, 2015. Web. 2 Feb. 2015. <http://www.eia.gov/electricity/>.

Bibliography

"SendMessage Function." *Windows Dev Center*. Microsoft, 2014. Web. 10 Nov. 2014. <https://msdn.microsoft.com/en-us/library/windows/desktop/ms644950(v=vs.85).aspx>.

"SetSuspendState Function." *Windows Dev Center*. Microsoft, 2015. Web. 14 Jan. 2015. <https://msdn.microsoft.com/en-us/library/windows/desktop/aa373201(v=vs.85).aspx>.

"Top Ten Countries with Highest number of PCs." *Maps of World*. Compare Infobase, 2015. Web. 2 Feb. 2015. <http://www.mapsofworld.com/world-top-ten/world-top-ten-personal-computers-users-map.html>.

"What Is Arduino." *Arduino*. Arduino, n.d. Web. 11 Nov. 2014. <http://arduino.cc/en/Guide/Introduction>.

"WM_SYSCOMMAND Message." *Windows Dev Center*. Microsoft, 2014. Web. 10 Nov. 2014. <https://msdn.microsoft.com/en-us/library/windows/desktop/ms646360(v=vs.85).aspx>.