```
/*************************************************************
File name: Hudson's Dandelion Detector
Version:  34
Description: This version everything works! Dandelion_Detector_34
should be added to GitHub.

What works in this code:
2 stepper motors - for claw snipping of dandelions.
LCD - works to display debug messages to Dandy Detector.  Serial
monitor to PC also outputs debug messages.
Pixy2 - works with 2 programmed signatures for white or yellow
dandelion heads.
2 Detector LEDs - white and yellow which are placed on custom PCB.
Main motor driver - allows driving of robot.
Ultrasonic Sensor - determines if any obstocales in way of robot.  Ie
trees, pets, humans, etc.
Photoresistor - determine if sunlight is sufficient to operate.
Dandelion heads will open in sunlight and close at night.
Fan - to run only on pixy2 detection of white seeded dandelion heads to
blow into collection container.
GPS - to keep robot used for geolocation and to keep robot running in
correct geofenced area. Ie a person's back or front yard.
Driving Algorithm implemented - Turns left or right if ultrasonic
sensor detects an object in front of it.
Sunlight Detection Algorithm implemented - Only operates dandelion
detector when their is sufficient sunlight to allow the dandelions to
open in yellow or white.
Pixy2 Dandelion Signature and algorithm - Pixy2 detects white dandelion
or yellow dandelions via PixyII signature.  If yellow head cuts with
claw function.  If white head cuts and runs fan
  to collect dandelion seeds in removable container.

E-mail:kinghut25@gmail.com
Author: Hudson Jantzi
Date: 2021/12/11

Website links
/Ийttps://docs.pixycam.com/wiki/doku.php?
```

```
id=wiki:v2:hooking_up_pixy_to_a_microcontroller_-28like_an_arduino-29
//https://docs.arduino.cc/hardware/mega-2560
*****************************************************/

//Includes LCD screen libraries
#include<Wire.h>
#include<LiquidCrystal_I2C.h>

// Include for Pixy library.
#include <Pixy2.h>

// This is the main Pixy object
Pixy2 pixy;

LiquidCrystal_I2C lcd(0x27,20,4);  // set the LCD address to 0x27 for a
16 chars and 2 line display

// Global Variable Definitions
// Left Side Stepper Motors for dandelion cutter.
int Pin0 = 33;  //definition digital 33 pins as pin to control the IN1
(ULN24L01)
int Pin1 = 35; //definition digital 35 pins as pin to control the IN2
(ULN24L01)
int Pin2 = 37; //definition digital 37 pins as pin to control the IN3
(ULN24L01)
int Pin3 = 39;  //definition digital 39 pins as pin to control the IN4
(ULN24L01)

// Right Side Stepper Motors for dandelion cutter.
int Pin4 = 32;  //definition digital 32 pins as pin to control the IN1
(ULN24L01)
int Pin5 = 34;  //definition digital 34 pins as pin to control the IN2
(ULN24L01)
int Pin6 = 36;  //definition digital 36 pins as pin to control the IN3
(ULN24L01)
int Pin7 = 38; //definition digital 38 pins as pin to control the IN4
(ULN24L01)

// Stepper motors for dandelion cutter step and speed variables.
```

```
int _step = 512;
int _speed = 1;

//Main Drivetrain Motor Pins - Drives Robot:
int motor1pin1 = 24;
int motor1pin2 = 22;
int motor2pin1 = 23;
int motor2pin2 = 25;

//Ultrasonic Setup
// Ultrasonic used to determine any object in front of dandelion
detector so robot won't hit/run into it.  Eg tree or fence in yard.
const int pingPin = 29; // Trigger Pin of Ultrasonic Sensor
const int echoPin = 28; // Echo Pin of Ultrasonic Sensor
bool obstacle_value;         // Store distance of obstacle in global
variable

//Photoresistor setup
//  Used to detect sunlight.  Dandelion heads will be opened when their
is sunlight.
const int pResistor = A14; // Photoresistor at Arduino analog pin A0
const int ledPin=14;      // Led pin at Arduino pin 13
int value;          // Store value from photoresistor (0-1023)

//Fan setup
// Fan used to blow dandelion white head seeds into container for
disposal.  Fan is only run when white dandelion detected.
// When yellow dandelion color detected by Pixy2 no need to run fan.
const int RELAY_PIN = A15;  // the Arduino pin, which connects to the
IN pin of relay

// GPS setup
// NEMA commands show GPS information such as latitude, longitude, GPS
or GNSS satellite constellation and number of satellites seen in open
sky.
#include<MicroNMEA.h>
#include <Wire.h>

//I2C communication parameters
```

```
#define DEFAULT_DEVICE_ADDRESS 0x3A
#define DEFAULT_DEVICE_PORT 0xFF
#define I2C_DELAY 1

#define RESET_PIN 7

#ifdef ARDUINO_SAM_DUE
#define DEV_I2C Wire1
#endif

#ifdef ARDUINO_ARCH_STM32
#define DEV_I2C Wire
#endif

#ifdef ARDUINO_ARCH_AVR
#define DEV_I2C Wire
#endif

// Refer to Stream devices by use
HardwareSerial& console = Serial;
TwoWire& gps = DEV_I2C;

//I2C read data structures
char buff[32];
int idx = 0;

//MicroNMEA library structures
char nmeaBuffer[100];
MicroNMEA nmea(nmeaBuffer, sizeof(nmeaBuffer));

bool ledState = LOW;
volatile bool ppsTriggered = false;

void ppsHandler(void);

////////////////////////////////////////////////////SETUP////////
////////////
// Main setup Function
// Initalizing and bringup of all HW features.
```

```
void setup()
{
  Serial.begin(115200);  // Starting Serial Terminal
 Serial.print("Starting...\n");
  pixy.init();

  // For LED initialization
  pinMode(16, OUTPUT);
  pinMode(15, OUTPUT);

  // initialize the lcd
  lcd.init();
  // Print a message to the LCD.
  lcd.begin(16,2);//Defining 16 columns and 2 rows of lcd display
  lcd.backlight();
  lcd.setCursor(1,0);
  lcd.print("Dandy Detector");
  lcd.setCursor(1,1);
  lcd.print("22 Science Fair");

  // Initialize the Stepper Motor to run custom dandelion claw cutter.
  pinMode(Pin0, OUTPUT); //Set digital 33 port mode, the OUTPUT for the
output
  pinMode(Pin1, OUTPUT); //Set digital 35 port mode, the OUTPUT for the
output
  pinMode(Pin2, OUTPUT); //Set digital 37 port mode, the OUTPUT for the
output
  pinMode(Pin3, OUTPUT); //Set digital 39 port mode, the OUTPUT for the
output

  pinMode(Pin4, OUTPUT); //Set digital 32 port mode, the OUTPUT for the
output
  pinMode(Pin5, OUTPUT); //Set digital 34 port mode, the OUTPUT for the
output
  pinMode(Pin6, OUTPUT); //Set digital 36 port mode, the OUTPUT for the
output
  pinMode(Pin7, OUTPUT); //Set digital 38 port mode, the OUTPUT for the
output
```

```
 // Main Drivetrain Motor Pins - Drives Robot:
 pinMode(motor1pin1, OUTPUT);
 pinMode(motor1pin2, OUTPUT);
 pinMode(motor2pin1, OUTPUT);
 pinMode(motor2pin2, OUTPUT);

 //Photoresistor Setup
  pinMode(ledPin, OUTPUT);  // Set ledPin - 13 pin as an output
  pinMode(pResistor, INPUT);// Set pResistor - A0 pin as an input
(optional)

 //Fan setup
 pinMode(RELAY_PIN, OUTPUT);

 // GPS setup
 console.begin(115200); // output to serial console
 gps.begin(); // gps starts/enables.

 pinMode(LED_BUILTIN, OUTPUT);
 digitalWrite(LED_BUILTIN, ledState);

 //Start the GPS module
 pinMode(RESET_PIN, OUTPUT);
 digitalWrite(RESET_PIN, HIGH);
 console.println("Resetting GPS module ...");
 gpsHardwareReset();
 console.println("... done");

 // Change the echoing messages to the ones recognized by the
MicroNMEA library
 sendCommand((char*)"$PSTMSETPAR,1231,0x00000042");
 sendCommand((char *)"$PSTMSAVEPAR");

 //Reset the device so that the changes could take plaace
 sendCommand((char *)"$PSTMSRR");

 delay(4000);

 //Reinitialize I2C after the reset
```

```
 gps.begin();

 //clear i2c buffer
 char c;
 idx = 0;
 memset(buff, 0, 32);
 do
  {
     if (idx == 0)
     {
        readI2C(buff);
        delay(I2C_DELAY);
     }
     c = buff[idx];
     idx++;
     idx %= 32;
  }
 while ((uint8_t) c != 0xFF);

 pinMode(2, INPUT);
attachInterrupt(digitalPinToInterrupt(2),ppsHandler,RISING);
}

void loop()
{
 Serial.print("Hudson's Dandelion Detector loop starting...\n");
  //digitalWrite(6, HIGH);  // turn on build in LED
  //digitalWrite(7, HIGH);  // turn on build in LED

 //Test photoresistor
Serial.print("Photoresistor/Sunshinetest...\n");

  // Print a message to the LCD.
 lcd.begin(16,2);//Defining 16 columns and 2 rows of lcd display
 lcd.backlight();
 lcd.setCursor(1,0);
 lcd.print("Check Sunlight");
 lcd.setCursor(1,1);
 lcd.print("22 Science Fair");
```

```
 SunshineCheck();
 delay(1000);
 SunshineCheck();

 //Test ultrasonic sensor
 Serial.print("Ultrasonic test on Dandelion Detector...\n");
  // Print a message to the LCD.
 lcd.begin(16,2);//Defining 16 columns and 2 rows of lcd display
 lcd.backlight();
 lcd.setCursor(1,0);
 lcd.print("Obstacle Check");
 lcd.setCursor(1,1);
 lcd.print("Clear to Drive");
 UltrasonicDistance();

//  Serial.print("Drive motor test on Dandelion Detector...\n");
 MoveForward();
 delay(500);
 Stop();

 // Run dandelion claw clipping.
 ClawRun();

 // If no obstacles drive forward\
}""  // Run dandelion claw clipping.
 ClawRun();
//  if (obstacle_value = true)
//    MoveRight();
//  else
//    MoveForward();
//
//    delay(500);
//    Stop();

 //Test Robot Movement
 Serial.print("Drive motor test on Dandelion Detector...\n");
 MoveForward();
 delay(1000);
```

```
  Stop();
//  delay(1000);
// MoveBackward();
//  delay(1000);
//  Stop();



  //Test fan
  FanOn();
  delay(2000);
  FanOff();
  // delay(2000);
//  FanOn();
//  delay(2000);
//  FanOff();

  //Test GPS scanning for satellite constellations in the sky.
  GPSScan();

  //*****************************MAIN BLOCK OF CODE CONDITION STATEMENT
IFDANDELLIONDETECTED*****************************************
  // Determines if white dandelions or yellow dandelions are present.
  // Only run dandelion detector if photoresistor detects sunshine
above the threshold limit.
  // If "Yellow" is detected by Pixy2 camera, stop driving, run custom
cutters to clip dandelion yellow head.
  // If "White" is detected by Pixy2 camera, stop driving, run custom
cutters to clip dandelion white head, then run fan to blow
  // white dandelion seeds into collector/container for disposal.
  // Pixy 2
  int i;
  uint16_t blocks;

while (1)
{
  blocks = pixy.ccc.getBlocks();

  if (blocks)
```

```
  {
   for (i=0; i<blocks;i++)
    {

   // If there are detect blocks, print them!
   // Open up Tools-> Serial Monitor to view.
   if (pixy.ccc.numBlocks)
    {
     Serial.print("Detected ");
    Serial.println(pixy.ccc.numBlocks);
     for (i=0; i<pixy.ccc.numBlocks; i++)
      {
       Serial.print("  block ");
       Serial.print(i);
       Serial.print(": ");
      pixy.ccc.blocks[i].print();

//   Code below determines if it is signature 1 or signature 2
detected by Pixy2 Camera.
//   If signature 1 is determined it turns the LED1 on and stepper
motor clockwise.
//   If signature 2 is determined it turns the LED2 on and stepper
motorcounterclockwise.
//   Signature 1 is white
//   Signature 2 is yellow
//
     // *** SIGNATURE 1 IF STATEMENT ***
     // WHITE DANDELION DETECTION
     if (pixy.ccc.blocks[i].m_signature == 1)
      {
     // do stuff for sig number one
     digitalWrite(16, HIGH);  // turn on build in LED for White
Dandelion Detection.
     //  delay (5000);  // Wait 5 seconds then turn off LED

    lcd.setCursor(0,0); //Defining position to write from first row,
first column .
    lcd.print("White Dandelion"); //You can write 16 Characters per
line .
```

```
 // Run Custom Dandelion Cutter
 // Print a message to the LCD.
lcd.begin(16,2);//Defining 16 columns and 2 rows of lcd display
 lcd.backlight();
 lcd.setCursor(0,0);
lcd.print("Scissors Cutting");
 lcd.setCursor(0,1);
lcd.print("White Dandelion");

 ClawRun();

 // Run the fan only if a white head
Serial.print("Fan running on Dandelion Detector...\n");
 // Print a message to the LCD.
lcd.begin(16,2);//Defining 16 columns and 2 rows of lcd display
 lcd.backlight();
 lcd.setCursor(1,0);
 lcd.print("Fan Running");
 lcd.setCursor(1,1);
lcd.print("Seeds Collected");
 FanOn();
 delay(2000);
 FanOff();
 delay(2000);

 //Test ultrasonic sensor after cut for next movement.
Serial.print("Ultrasonic test on Dandelion Detector...\n");
 // Print a message to the LCD.
lcd.begin(16,2);//Defining 16 columns and 2 rows of lcd display
 lcd.backlight();
 lcd.setCursor(1,0);
lcd.print("Obstacle Check");
 lcd.setCursor(1,1);
 lcd.print("Clear to Drive");
UltrasonicDistance();

 // If no obstacles drive forward, else turn right.
 if (obstacle_value = true)
```

```
      MoveRight();
     else
      MoveForward();

     delay(500);
     Stop();

     digitalWrite(16, LOW);  // turn off build in LED

      }
////////////////////////////////////////////////////////////////////////////
////////////////////////////////////
     // *** SIGNATURE 2 IF STATEMENT ***
     // YELLOW DANDELION DETECTION
     if (pixy.ccc.blocks[i].m_signature == 2)
      {
     // do stuff for sig number two
     digitalWrite(15, HIGH);  // turn on build in LED for Yellow
Dandelion Detection.
     // delay (5000);

    lcd.setCursor(0,0); //Defining positon to write from first row,
first column .
    lcd.print("Yellow Dandelion"); //You can write 16 Characters per
line .

     // Run Custom Dandelion Cutter
     // Print a message to the LCD.
    lcd.begin(16,2);//Defining 16 columns and 2 rows of lcd display
     lcd.backlight();
     lcd.setCursor(0,0);
    lcd.print("Scissors Cutting");
     lcd.setCursor(0,1);
    lcd.print("Yellow Dandelion");
     ClawRun();

    //Test ultrasonic sensor after cut for next movement.
   Serial.print("Ultrasonic test on Dandelion Detector...\n");
     // Print a message to the LCD.
```

```
    lcd.begin(16,2);//Defining 16 columns and 2 rows of lcd display
    lcd.backlight();
    lcd.setCursor(1,0);
    lcd.print("Obstacle Check");
    lcd.setCursor(1,1);
    lcd.print("Clear to Drive");
    UltrasonicDistance();

    // If no obstacles drive forward, else turn right.
    if (obstacle_value = true)
      MoveRight();
     else
      MoveForward();

     delay(500);
     Stop();

      digitalWrite(15, LOW);  // turn off build in LED

      }

   ////////////////////////////////////////////////////
//// *** SIGNATURE 3 IF STATEMENT ***
//      // NO DANDELION DETECTION
    if (pixy.ccc.blocks[i].m_signature != 1 || pixy.ccc.blocks[i].
m_signature !=2)
      {
     // do stuff for sig number three no dandelion detected.
     // Drive forward, checking for obstacles when no dandelion
detected.

    lcd.setCursor(0,0); //Defining positon to write from first row,
first column .
    lcd.print("Green Grass"); //You can write 16 Characters per line .

     // Run Custom Dandelion Cutter
     // Print a message to the LCD.
    lcd.begin(16,2);//Defining 16 columns and 2 rows of lcd display
    lcd.backlight();
```

```
    lcd.setCursor(0,0);
    lcd.print("Green Grass");
    lcd.setCursor(0,1);
    lcd.print("Find Dandelion");

    //Test ultrasonic sensor after cut for next movement.
   Serial.print("Ultrasonic test on Dandelion Detector...\n");
    // Print a message to the LCD.
    lcd.begin(16,2);//Defining 16 columns and 2 rows of lcd display
    lcd.backlight();
    lcd.setCursor(1,0);
    lcd.print("Obstacle Check");
    lcd.setCursor(1,1);
    lcd.print("Clear to Drive");
    UltrasonicDistance();

    // If no obstacles drive forward, else turn right.
    if (obstacle_value = true)
      MoveRight();
     else
      MoveForward();
      delay(500);
     Stop();

     }

     }
     }
   }

  }
}
}

// Both stepper motors speed function
void Speed(int stepperspeed)//set Stepper speed
{
   _speed = 15 - stepperspeed;
   if( _speed<1){
```

```
    _speed = 1;
   }
  if( _speed>15){
    _speed = 15;
   }
}

// Stepper motor 1 step function
void Step(int _step)//Stepper motor rotation
{
 if(_step>=0){  // Stepper motor forward
  for(int i=0;i<_step;i++){
    setStep(1, 0, 0, 1);
    delay(_speed);
    setStep(1, 0, 0, 0);
    delay(_speed);
    setStep(1, 1, 0, 0);
    delay(_speed);
    setStep(0, 1, 0, 0);
    delay(_speed);
    setStep(0, 1, 1, 0);
    delay(_speed);
    setStep(0, 0, 1, 0);
    delay(_speed);
    setStep(0, 0, 1, 1);
    delay(_speed);
    setStep(0, 0, 0, 1);
    delay(_speed);
   }
 }else{ // Stepper motor backward
   for(int i=_step;i<0;i++){
    setStep(0, 0, 0, 1);
    delay(_speed);
    setStep(0, 0, 1, 1);
    delay(_speed);
    setStep(0, 0, 1, 0);
    delay(_speed);
    setStep(0, 1, 1, 0);
    delay(_speed);
```

```
      setStep(0, 1, 0, 0);
      delay(_speed);
      setStep(1, 1, 0, 0);
      delay(_speed);
      setStep(1, 0, 0, 0);
      delay(_speed);
      setStep(1, 0, 0, 1);
      delay(_speed);
     }
    }
}

//SetStep function for motor 1
void setStep(int a, int b, int c, int d)
{
   digitalWrite(Pin0, a);
   digitalWrite(Pin1, b);
   digitalWrite(Pin2, c);
   digitalWrite(Pin3, d);
}

// Stepper motor 2 step function
void Step2(int _step2)//Stepper motor rotation
{
  if(_step2>=0){  // Stepper motor forward
    for(int i=0;i<_step2;i++){
      setStep2(1, 0, 0, 1);
      delay(_speed);
      setStep2(1, 0, 0, 0);
      delay(_speed);
      setStep2(1, 1, 0, 0);
      delay(_speed);
      setStep2(0, 1, 0, 0);
      delay(_speed);
      setStep2(0, 1, 1, 0);
      delay(_speed);
      setStep2(0, 0, 1, 0);
      delay(_speed);
      setStep2(0, 0, 1, 1);
```

```
     delay(_speed);
     setStep2(0, 0, 0, 1);
     delay(_speed);
    }
  }else{ // Stepper motor backward
    for(int i=_step2;i<0;i++){
     setStep2(0, 0, 0, 1);
     delay(_speed);
     setStep2(0, 0, 1, 1);
     delay(_speed);
     setStep2(0, 0, 1, 0);
     delay(_speed);
     setStep2(0, 1, 1, 0);
     delay(_speed);
     setStep2(0, 1, 0, 0);
     delay(_speed);
     setStep2(1, 1, 0, 0);
     delay(_speed);
     setStep2(1, 0, 0, 0);
     delay(_speed);
     setStep2(1, 0, 0, 1);
     delay(_speed);
    }
   }
}

//SetStep function for motor 2
void setStep2(int e, int f, int g, int h)
{
   digitalWrite(Pin4, e);
   digitalWrite(Pin5, f);
   digitalWrite(Pin6, g);
   digitalWrite(Pin7, h);
}

//Move Dandelion Picker Chassis Forward
voidMoveForward()
{
 digitalWrite(motor1pin1, HIGH);
```

```
 digitalWrite(motor1pin2, LOW);

 digitalWrite(motor2pin1, HIGH);
 digitalWrite(motor2pin2, LOW);
  //delay(1000);

}

//Move Dandelion Picker Chassis Backward
voidMoveBackward()
{
 digitalWrite(motor1pin1, LOW);
 digitalWrite(motor1pin2, HIGH);

 digitalWrite(motor2pin1, LOW);
 digitalWrite(motor2pin2, HIGH);
  //delay(1000);
}

//Move Dandelion Picker Chassis Right
voidMoveRight()
{
 digitalWrite(motor1pin1, HIGH);
 digitalWrite(motor1pin2, LOW);

 digitalWrite(motor2pin1, LOW);
 digitalWrite(motor2pin2, HIGH);
  //delay(1000);
}

//Move Dandelion Picker Chassis Right
voidMoveLeft()
{
 digitalWrite(motor1pin1, LOW);
 digitalWrite(motor1pin2, HIGH);

 digitalWrite(motor2pin1, HIGH);
 digitalWrite(motor2pin2, LOW);
  //delay(1000);
```

```
}

//Stop Dandelion Picker Chassis Movement
void Stop()
{
 digitalWrite(motor1pin1, LOW);
 digitalWrite(motor1pin2, LOW);

 digitalWrite(motor2pin1, LOW);
 digitalWrite(motor2pin2, LOW);
 // delay(1000);
}

//Ultrasonic Distance Function to determine if any obstacle blocking
robot path
// obstacle_value variable tells if obstacle in way.
voidUltrasonicDistance()
{
  long duration, inches, cm;
  pinMode(pingPin, OUTPUT);
  digitalWrite(pingPin, LOW);
  delayMicroseconds(2);
  digitalWrite(pingPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(pingPin, LOW);
  pinMode(echoPin, INPUT);
  duration = pulseIn(echoPin, HIGH);
  inches = microsecondsToInches(duration);
  cm = microsecondsToCentimeters(duration);
  Serial.print(inches);
  Serial.print("in, ");
  Serial.print(cm);
  Serial.print("cm");
  Serial.println();
  if (cm >20)
    obstacle_value = true;
  else
    obstacle_value = false;
    delay(100);
```

```
}
//Converts Microseconds to inches
longmicrosecondsToInches(longmicroseconds){
   return microseconds / 74 / 2;
}
//Converts Microseconds to centimeters
longmicrosecondsToCentimeters(longmicroseconds){
   return microseconds / 29 / 2;
}

//Photoresistor Sunlight Check Function
// Threshold set for 900.
// If sunlight is bright, larger than threshold enable dandelion
detector.
voidSunshineCheck(){
 value = analogRead(pResistor);
 Serial.print ("Photoresistor Value:\n");
 Serial.print (value);
  //You can change value "900"
  if (value > 100) {
    digitalWrite(ledPin, LOW);  //Turn led off
    lcd.begin(16,2);//Defining 16 columns and 2 rows of lcd display
    lcd.backlight();
    lcd.setCursor(1,0);
    lcd.print("No Sunlight");
    lcd.setCursor(1,1);
    lcd.print("Dandy closed");
  }
  else{
    digitalWrite(ledPin, HIGH); //Turn led on
    lcd.begin(16,2);//Defining 16 columns and 2 rows of lcd display
    lcd.backlight();
    lcd.setCursor(1,0);
    lcd.print("Sunlight OK");
    lcd.setCursor(1,1);
    lcd.print("Run DandyDet");
  }

 delay(500); //Small .5sec delay
```

```
}

//Fan on function
// Used to collect white dandelion sead heads only.
void FanOn(){
   digitalWrite(RELAY_PIN, HIGH); // turn on fan
}

//Fan off function
void FanOff(){
  digitalWrite(RELAY_PIN, LOW);  // turn off fan 5 seconds
}


// GPS functions
voidppsHandler(void)
{
  ppsTriggered = true;
}

// GPS hardware reset function
voidgpsHardwareReset()
{
   //reset the device
  digitalWrite(RESET_PIN, LOW);
   delay(50);
  digitalWrite(RESET_PIN, HIGH);

   //wait for reset to apply
   delay(2000);

}

//Read 32 bytes from I2C for GPS
void readI2C(char *inBuff)
{
 gps.beginTransmission(DEFAULT_DEVICE_ADDRESS);
  gps.write((uint8_t) DEFAULT_DEVICE_PORT);
```

```
 gps.endTransmission(false);
 gps.requestFrom((uint8_t)DEFAULT_DEVICE_ADDRESS, (uint8_t) 32);
   int i = 0;
  while (gps.available())
   {
     inBuff[i]= gps.read();
      i++;
   }
}

//Send a NMEA command via I2C for GPS
void sendCommand(char *cmd)
{
 gps.beginTransmission(DEFAULT_DEVICE_ADDRESS);
  gps.write((uint8_t) DEFAULT_DEVICE_PORT);
  MicroNMEA::sendSentence(gps, cmd);
  gps.endTransmission(true);
}

// GPS scanning sky for conseltations.  Need to have a clear view of
the sky to get a fix.
// prints GPS NEMA messages to serial console.
void GPSScan()
{
  //If a message is recieved print all the informations
  if (ppsTriggered)
   {
     ppsTriggered = false;
     ledState = !ledState;
    digitalWrite(LED_BUILTIN, ledState);

    // Output GPS information from previous second
    console.print("Valid fix: ");
    console.println(nmea.isValid() ? "yes" : "no");

    console.print("Nav. system: ");
    if (nmea.getNavSystem())
      console.println(nmea.getNavSystem());
     else
```

```
    console.println("none");

 console.print("Num. satellites: ");
console.println(nmea.getNumSatellites());

 console.print("HDOP: ");
console.println(nmea.getHDOP()/10., 1);

 console.print("Date/time: ");
console.print(nmea.getYear());
 console.print('-');
console.print(int(nmea.getMonth()));
 console.print('-');
console.print(int(nmea.getDay()));
 console.print('T');
console.print(int(nmea.getHour()));
 console.print(':');
console.print(int(nmea.getMinute()));
 console.print(':');
console.println(int(nmea.getSecond()));

 long latitude_mdeg = nmea.getLatitude();
 long longitude_mdeg = nmea.getLongitude();
 console.print("Latitude (deg): ");
 console.println(latitude_mdeg / 1000000., 6);

 console.print("Longitude (deg): ");
 console.println(longitude_mdeg / 1000000., 6);

  long alt;
 console.print("Altitude (m): ");
 if (nmea.getAltitude(alt))
    console.println(alt / 1000., 3);
  else
    console.println("not available");

 console.print("Speed: ");
 console.println(nmea.getSpeed() / 1000., 3);
 console.print("Course: ");
```

```
    console.println(nmea.getCourse() / 1000., 3);
    console.println("----------------------");
     nmea.clear();
   }

  //While the message isn't complete
  while (!ppsTriggered )
   {
      char c ;
      if (idx == 0)
      {
        readI2C(buff);
        delay(I2C_DELAY);
      }
      //Fetch the character one by one
      c = buff[idx];
      idx++;
      idx %= 32;
      //If we have a valid character pass it to the library
      if ((uint8_t) c != 0xFF)
      {
        console.print(c);
        nmea.process(c);
      }
   }
delay(2000);
}

// Function to run the dandelion claw to clip,cut, snip dandelion heads.
void ClawRun() {

 Serial.print("Claw stepper motor test on Dandelion Detector...\n");
 // First/Left Stepper motor Test
 // 128 first, -128 second
 Speed(15);//Stepper motor speed = 15 fast (note:speed from 1 to 15)
 Step(-128);//Stepper motor forward 512 steps ---- 360 angle
 // delay(2000);// delay 2S  // removed 2s delays as was taking too
long to clip dandelion heads.
```

```
 // Second/Right stepper motor test
 // 128 first, -128 second
 Speed(15);//Stepper motor speed = 15 fast (note:speed from 1 to 15)
 Step2(128);//Stepper motor forward 512 steps ---- 360 angle
 //delay(2000);// delay 2S

 Speed(15); //Stepper motor speed = 1 slow  (note:speed from 1 to 15)
 Step(128);//Stepper motor backward 512 steps ---- 360 angle
 //delay(2000);//delay 2S

 Speed(15);//Stepper motor speed = 15 fast (note:speed from 1 to 15)
 Step2(-128);//Stepper motor forward 512 steps ---- 360 angle
 //delay(2000);// delay 2S

}
```