Audio Trigger Via PIR Sensor

June 2020

1 DISCLAIMER

You are hereby authorized to view, copy, print, and distribute the materials from this document or the attached drawings subject to the following conditions:

- No liability in respect of any of the contents of such publications or use thereof is accepted and no warranties expressed or implied are made in relation thereto.

Disclaimer

ANY INFORMATION CONTAINED IN THIS DOCUMENT OR THE ATTACHED DRAWINGS ARE PROVIDED [AS IS] FOR YOUR INFORMATIONAL PURPOSES ONLY, WITHOUT WARRANTY OF ANY KIND, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. THE AUTHOR OF THIS DOCUMENT AND ATTACHED DRAWINGS WILL IN NO EVENT BE LIABLE FOR ANY DIRECT, INDIRECT OR PUNITIVE DAMAGE OF ANY KIND THAT RESULTS FROM THE USE OF OR INABILITY TO USE THIS DOCUMENT OR THE ATTACHED DRAWINGS, INCLUDING WITHOUT LIMITATION, LOSS OF PROFITS, GOODWILL, OR BUSINESS INTERRUPTION. THE AUTHOR OFFERS NO ASSURANCES, WARRANTIES OR GUARANTEES AS TO THE VALIDITY OR COMPLETENESS OF THIS INFORMATION. INFORMATION WILL BE CHANGED, UPDATED AND DELETED WITHOUT NOTICE.

2 Overview

The intent of this document is to provide information on how to implement audio playback based on a PIR sensor becoming activated.

The project materials utilized are summarized below and a detailed list of the components for purchasing the materials is listed in section 3 of this document.

- (1) Arduino Nano
- (1) HiLetgo HC-SR501 PIR Infrared Sensor
- (1) HiLetgo YX5300 UART Control Serial MP3 Music Player Module
- (1) MicroSd card
- (1) Panel Mount 2.1mm DC barrel jack (Optional)
- (1) Fuse (1 amp)
- (1) 5 VDC Wall Power Adapter (2000mA) – Connector 2.1mm plug
- (1) Audio Amplifier Board, DROK 5W+5W Mini Amplifier Board
- (1) Speaker
- (1) 3.5mm Mini Stereo Audio Cable with Plugs (M/M)

3 Parts List

The following list details all of the electrical components used in the system.

Description	Part Number	QTY	Purchased From
Panel Mount 2.1mm DC barrel jack	ID:1612	1	Adafruit.com
Audio Amplifier Board, DROK 5W+5W Mini Amplifier Board PAM8406 DC 5V Digital Stereo Power Amp 2.0 Dual Channel Class D Amplify Module for Speaker Sound System DIY	-	1	amazon.com https://www.amazon.com/gp/product/B077MKQJW2
HiLetgo YX5300 UART Control Serial MP3 Music Player Module For Arduino/AVR/ARM/PIC	-	1	amazon.com https://www.amazon.com/gp/product/B0725RHR4D
HiLetgo 3pcs HC-SR501 PIR Infrared Sensor Human Body Infrared Motion Module for Arduino Raspberry Pi	-	1	amazon.com https://www.amazon.com/gp/product/B07KZW86YR
Gikfun 2" 4Ohm 3W Full Range Audio Speaker Stereo Woofer Loudspeaker for Arduino (Pack of 2pcs) EK1725	-	1	amazon.com https://www.amazon.com/gp/product/B01CHYIU26
iMBAPrice 5V DC Wall Power Adapter UL Listed Power Supply (5-Feet, 5V 2A(2000mA))	-	1	amazon.com https://www.amazon.com/gp/product/B00GUO5WUI
Nano V3.0, Nano Board ATmega328P 5V 16M Micro-Controller Board Compatible with Arduino IDE (Nano x 3 with USB Cable)	-	1	amazon.com https://www.amazon.com/LAFVIN-Board-ATmega328P-Micro-Controller-Arduino/dp/B07G99NNXL

Description	Part Number	QTY	Purchased From
SanDisk Ultra 32GB MicroSDHC UHS-I Card with Adapter - 98MB/s U1 A1 - SDSQUAR-032G-GN6MA	-	1	amazon.com https://www.amazon.com/SanDisk-Ultra-microSDXC-Memory-Adapter/dp/B073JWXGNT
Tripp Lite (P312-001-2RA) 3.5mm Mini Stereo Audio Cable with Two Right Angle Plugs (M/M), 1-ft	-	1	amazon.com https://www.amazon.com/gp/product/B00M5FKEUE

4 Electrical Summary and Schematics

The wiring schematics are provided at the end of this document. A brief summary of the electrical circuit is as follow:

Audio Playback

When the PIR sensor detects motion, the Arduino will detect this on Pin 5 and send 1 of 15 different serial commands to the audio playback board. The audio playback board has a MicroSD slot on it for the music files. To use the audio playback board, you must do the following:

- Format the card as FAT32
- Create a directory on the card named 01 (that's zero one)
- Place your audio files in the 01 directory and their names must be 001.wav, 002.wav, 003.wav …. 015.wav. I think you can also use .mp3 files as well but the location and naming of the files is critical. The audio files must be 3-digit prefixes and numerically based as shown above.

Once the PIR sensor is triggered it will play file 001.wav and wait ten seconds from the time the audio file starts playing before it will look for another PIR trigger. The 10 second duration can be changed in the Arduino code but the duration must not be less then the duration of your longest audio file. If the duration time is too small, the audio will get cutoff. On the next trigger it will play file 002.wav. This process will continue until audio file 015.wav is played. On the next trigger after playing 015.wav, the playback will start back at audio file 001.wav. If you only have a few files that you want to cycle through, just comment out the code as required in the Arduino.

5 Arduino Nano Code

```
/*
John Guarnero 2020
Nano Trigger Audio when PIR active
*/

// Import the required libraries
#include <Wire.h>              // For I2C comm, but needed for not getting compile error
#include <SoftwareSerial.h>

// The software serial connections:
// * RX is digital pin 12 (connect to TX of other device)
// * TX is digital pin 11 (connect to RX of other device)
//
//Since the audio player uses serial communication, we are using the software serial library because
//the Nano only has its programming port as a serial port and we want to be connected for programming while connected to the audio player.

SoftwareSerial mySerial(12, 11); // RX, TX


int MotionPresent;
int PIRSensor;
int PlayMotionMessage = 1;
const int Pin5 = 5; //PIR Input

unsigned long currentMillis;
unsigned long previousMillis = 0;
unsigned long PlayStartMillis = 0;


//Define the messages serial command to play, This is the serial data you must send to the audio player to play the file
//Note: The micro SD card should be formatted as FAT 32 and have a folder named '01'. You put your files in this folder.
//The files must be named 001.wav, 002.wav, 003.wav .....
//I think MP3 will work as well but I know WAV files work because that what I use

byte message01[] = {0x7E, 0xFF, 0x06, 0x0F, 0x00, 0x01, 0x01, 0xEF }; // File 001
byte message02[] = {0x7E, 0xFF, 0x06, 0x0F, 0x00, 0x01, 0x02, 0xEF }; // File 002
byte message03[] = {0x7E, 0xFF, 0x06, 0x0F, 0x00, 0x01, 0x03, 0xEF }; // File 003
```

```
byte message04[] = {0x7E, 0xFF, 0x06, 0x0F, 0x00, 0x01, 0x04, 0xEF }; // File 004
byte message05[] = {0x7E, 0xFF, 0x06, 0x0F, 0x00, 0x01, 0x05, 0xEF }; // File 005
byte message06[] = {0x7E, 0xFF, 0x06, 0x0F, 0x00, 0x01, 0x06, 0xEF }; // File 006
byte message07[] = {0x7E, 0xFF, 0x06, 0x0F, 0x00, 0x01, 0x07, 0xEF }; // File 007
byte message08[] = {0x7E, 0xFF, 0x06, 0x0F, 0x00, 0x01, 0x08, 0xEF }; // File 008
byte message09[] = {0x7E, 0xFF, 0x06, 0x0F, 0x00, 0x01, 0x09, 0xEF }; // File 009
byte message10[] = {0x7E, 0xFF, 0x06, 0x0F, 0x00, 0x01, 0x0A, 0xEF }; // File 010
byte message11[] = {0x7E, 0xFF, 0x06, 0x0F, 0x00, 0x01, 0x0B, 0xEF }; // File 011
byte message12[] = {0x7E, 0xFF, 0x06, 0x0F, 0x00, 0x01, 0x0C, 0xEF }; // File 012
byte message13[] = {0x7E, 0xFF, 0x06, 0x0F, 0x00, 0x01, 0x0D, 0xEF }; // File 013
byte message14[] = {0x7E, 0xFF, 0x06, 0x0F, 0x00, 0x01, 0x0E, 0xEF }; // File 014
byte message15[] = {0x7E, 0xFF, 0x06, 0x0F, 0x00, 0x01, 0x0F, 0xEF }; // File 015

////////////////////////////////////////////////////////////////////////////////////////////////
void setup()
////////////////////////////////////////////////////////////////////////////////////////////////
{

// initialize the pushbutton pin as an input:



  pinMode(Pin5, INPUT); // PIR Sensor Input Pinmode


  // set the data rate for serial ports
  Serial.begin(9600);   // set the data rate for the built in serial port
  mySerial.begin(9600); // set the data rate for the SoftwareSerial port

  Wire.begin();

} // End Setup


////////////////////////////////////////////////////////////////////////////////////////////////
void loop() // run the code in this loop forever
////////////////////////////////////////////////////////////////////////////////////////////////
//Note: since we do not use delays in the program and use the 'millis()', you can just add
what you want for LEDs and such to the area below 'Void Loop()'
{

PIRSensor = digitalRead(Pin5);
// Serial.println (PIRSensor);  //This is here for testing
 //Serial.println (MotionPresent); //This is here for testing
```

```
 currentMillis = millis();   //millis is used to avoid using a delay command in the code

if (PIRSensor == 1 &&  MotionPresent == 0)
{
 PlayMotionAudio();
 }


 if (PIRSensor == 0 && currentMillis - PlayStartMillis > 10000)  //PIR Sensor is off and
10 seconds have elpased since start of playback.  The time must be > than the longest audio
file duration.
 {
  MotionPresent = 0;
 }


} //End of Void Loop


////////////////////////////////////////////////////////////////////////////////////////////
void PlayMotionAudio() //Subroutine For playing Audio when someone approaches
////////////////////////////////////////////////////////////////////////////////////////////
//This will play a single audio file when the PIR is triggered. Upon subsequent triggers, the
audio files will be played back in the order of file 001 to file 015 and then start back at file
001.
//If you have less audio files, just comment out the 'If' section for the audio file, be sure to
change the last PlayMotionMessage = xx; to PlayMotionMessage = 1;
{


//Play Message Due To Motion
if (MotionPresent == 0 && PlayMotionMessage == 1)
{
  PlayMotionMessage = 2;
  PlayStartMillis = millis();
  mySerial.write(message01, sizeof(message01)); // File 001
  MotionPresent = 1;// Set Motion present so you dont retrigger audio
  }

  if (PlayMotionMessage == 2  &&  MotionPresent == 0)
{
  PlayMotionMessage = 3;
  PlayStartMillis = millis();
  mySerial.write(message02, sizeof(message02)); // File 002
```

```
 MotionPresent = 1;// Set Motion present so you dont retrigger audio
 }

 if (PlayMotionMessage == 3  &&  MotionPresent == 0)
{
 PlayMotionMessage = 4;
 PlayStartMillis = millis();
 mySerial.write(message03, sizeof(message03)); // File 003
 MotionPresent = 1;// Set Motion present so you dont retrigger audio
 }


 if (PlayMotionMessage == 4  &&  MotionPresent == 0)
{
 PlayMotionMessage = 5;
 PlayStartMillis = millis();
 mySerial.write(message04, sizeof(message04)); // File 004
 MotionPresent = 1;// Set Motion present so you dont retrigger audio
 }


  if (PlayMotionMessage == 5  &&  MotionPresent == 0)
{
 PlayMotionMessage = 6;
 PlayStartMillis = millis();
 mySerial.write(message05, sizeof(message05)); // File 005
 MotionPresent = 1;// Set Motion present so you dont retrigger audio
 }


  if (PlayMotionMessage == 6  &&  MotionPresent == 0)
{
 PlayMotionMessage = 7;
 PlayStartMillis = millis();
 mySerial.write(message06, sizeof(message06)); // File 006
 MotionPresent = 1;// Set Motion present so you dont retrigger audio
 }

  if (PlayMotionMessage == 7  &&  MotionPresent == 0)
{
 PlayMotionMessage = 8;
 PlayStartMillis = millis();
 mySerial.write(message07, sizeof(message07)); // File 007
 MotionPresent = 1;// Set Motion present so you dont retrigger audio
 }
```

```
   if (PlayMotionMessage == 8  &&  MotionPresent == 0)
{
 PlayMotionMessage = 9;
 PlayStartMillis = millis();
 mySerial.write(message08, sizeof(message08));  // File 008
 MotionPresent = 1;// Set Motion present so you dont retrigger audio
 }

   if (PlayMotionMessage == 9  &&  MotionPresent == 0)
{
 PlayMotionMessage = 10;
 PlayStartMillis = millis();
 mySerial.write(message09, sizeof(message09));  // File 009
 MotionPresent = 1;// Set Motion present so you dont retrigger audio
 }

   if (PlayMotionMessage == 10  &&  MotionPresent == 0)
{
 PlayMotionMessage = 11;
 PlayStartMillis = millis();
 mySerial.write(message10, sizeof(message10)); // File 010
 MotionPresent = 1;// Set Motion present so you dont retrigger audio
 }


   if (PlayMotionMessage == 11  &&  MotionPresent == 0)
{
 PlayMotionMessage = 12;
 PlayStartMillis = millis();
 mySerial.write(message11, sizeof(message11)); // File 011
 MotionPresent = 1;// Set Motion present so you dont retrigger audio
 }


   if (PlayMotionMessage == 12  &&  MotionPresent == 0)
{
 PlayMotionMessage = 13;
 PlayStartMillis = millis();
 mySerial.write(message12, sizeof(message12)); // File 012
 MotionPresent = 1;// Set Motion present so you dont retrigger audio
 }


   if (PlayMotionMessage == 13  &&  MotionPresent == 0)
```

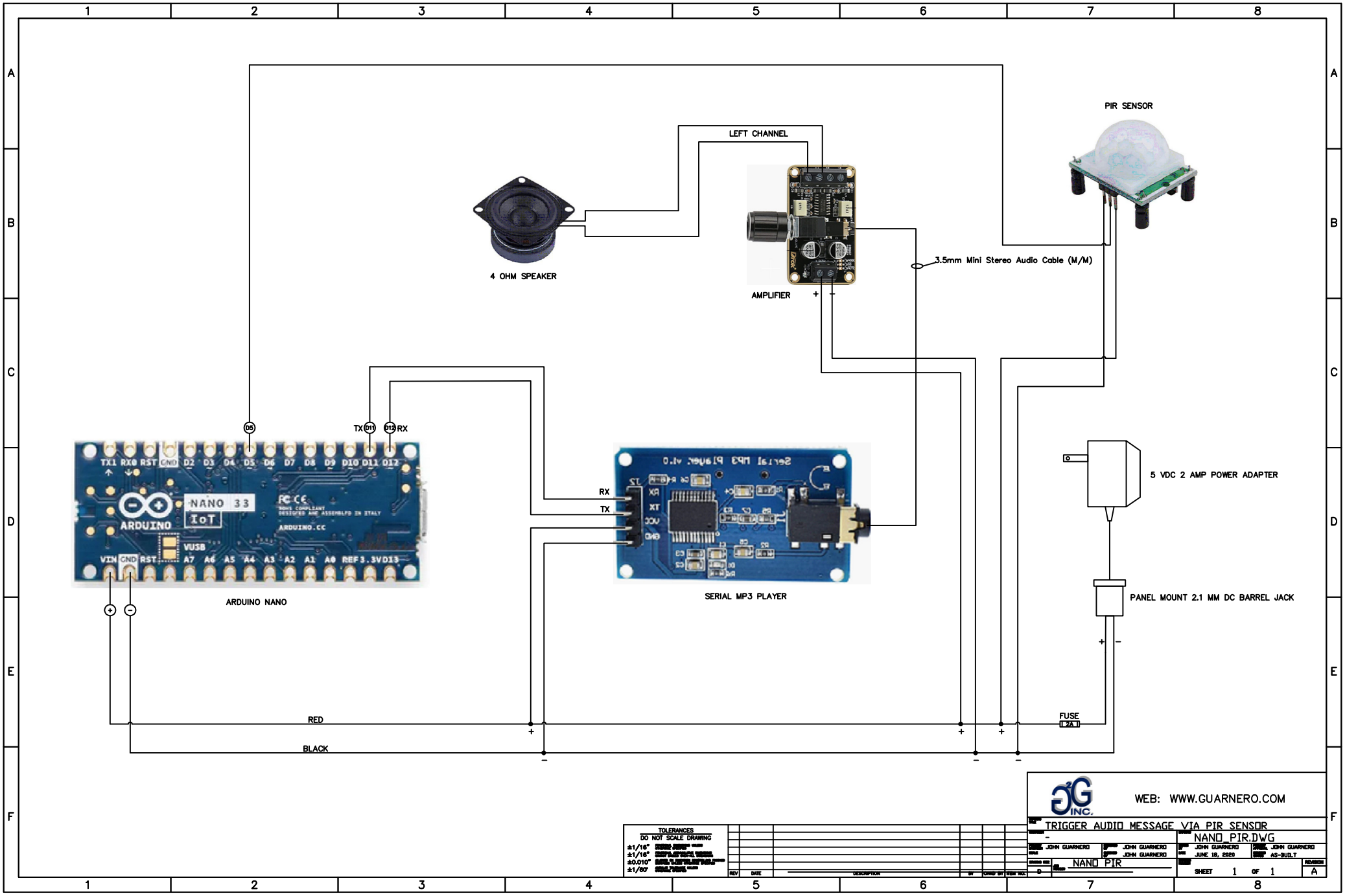
```
{
 PlayMotionMessage = 14;
 PlayStartMillis = millis();
 mySerial.write(message13, sizeof(message13)); // File 013
 MotionPresent = 1;// Set Motion present so you dont retrigger audio
 }


  if (PlayMotionMessage == 14  &&  MotionPresent == 0)
{
 PlayMotionMessage = 15;
 PlayStartMillis = millis();
 mySerial.write(message14, sizeof(message14)); // File 014
 MotionPresent = 1;// Set Motion present so you dont retrigger audio
 }

  if (PlayMotionMessage == 15  &&  MotionPresent == 0)
{
 PlayMotionMessage = 1;
 PlayStartMillis = millis();
 mySerial.write(message15, sizeof(message15)); // File 015
 MotionPresent = 1;// Set Motion present so you dont retrigger audio
 }


} //End of Void PlayMotionAudio
```

6 Electrical Diagram


PIR SENSOR
LEFT CHANNEL
4 OHM SPEAKER
AMPLIFIER
3.5mm Mini Stereo Audio Cable (M/M)
5 VDC 2 AMP POWER ADAPTER
PANEL MOUNT 2.1 MM DC BARREL JACK
ARDUINO NANO
SERIAL MP3 PLAYER
RX
TX
RED
BLACK
FUSE
2A T
WEB: WWW.GUARNERO.COM
TRIGGER AUDIO MESSAGE VIA PIR SENSOR
NANO_PIR.DWG
NANO PIR
SHEET 1 OF 1
A