

	#include	AudioConnectio		waveform2.amp	if(Pstate == 1){
#include	<wire.h></wire.h>	n	pinMode(Pbutto	litude(level*0.05)	
<eeprom.h></eeprom.h>	#include	patchCord1(wa	n, INPUT);	;	numberOfPPres
#include <onewire.h></onewire.h>	<spi.h></spi.h>	veform1, 0,	pinMode(Led,	waveform3.freq	ses += 1;
<onewire.n></onewire.n>	#include <sd.h></sd.h>	mixer1, 0); AudioConnectio	OUTPUT);	uency(tempFreq	Pcount = numberOfPPres
OneWire ds(20);	#include	n	Serial.begin(960	); waveform3.amp	ses % 2;
Offervire d3(20),	<serialflash.h></serialflash.h>	patchCord2(wa	0);	litude(level);	// //
int Button = 14;	Cochail lash.ii>	veform2, 0,	0),	waveform4.freq	Serial.println(Pc
int Batton = 14,	#include	mixer1, 1);	//audio	uency(pitchFreq	ount);
int	<audio.h></audio.h>	AudioConnectio	// dd d.o	);	PtriggerState
numberOfBPres	#include	n		waveform4.amp	= Pstate;
ses = 0;	<wire.h></wire.h>	patchCord3(sin	AudioMemory(1	litude(level);	,
int Bcount;	#include	e_fm1, 0,	0);	sine_fm1.freque	if (Pcount ==
int triggerState	<spi.h></spi.h>	mixer1, 2);	waveform1.begi	ncy(yawFreq);	0){
= 0;	#include	AudioConnectio	n(WAVEFORM_	sine_fm1.amplit	
int Led = $15$ ;	<sd.h></sd.h>	n	TRIANGLE);	ude(level);	Serial.println("St
int Pbutton =	#include	patchCord4(wa	waveform2.begi		op");
17;	<serialflash.h></serialflash.h>	veform4,	n(WAVEFORM_		
int Pstate = 0;		sine_fm1);	SQUARE);	// record toggle	Serial.println(rea
int PtriggerState	#include	AudioConnectio	waveform3.begi	Bstate =	dings[100]);
= 0;	<audio.h></audio.h>	n	n(WAVEFORM_	digitalRead(Butt	} :f(D==:::=t
int numberOfPPres	#include <wire.h></wire.h>	patchCord5(mix er1, 0, filter1, 0);	SINE); waveform4.begi	on);	if(Pcount ==
ses = 0;	#include	AudioConnectio	n(WAVEFORM_	if (Bstate !=	1){
int Pcount;	<spi.h></spi.h>	n	SINE);	triggerState){	Serial.println("Pl
int	#include	patchCord6(wa	pinMode(5,	if(Bstate == 1){	ay");
readings[100];	<sd.h></sd.h>	veform3, 0,	OUTPUT);	11(DState == 1)(	}
int Pfreq = 0;	#include	filter1, 1);	digitalWrite(5,HI	numberOfBPres	}else{
int	<serialflash.h></serialflash.h>	AudioConnectio	GH);	ses += 1;	PtriggerState
Areading[100];		n	delay(10);	Bcount =	= Pstate;
	// GUltool:	patchCord7(filte		numberOfBPres	}
//int Temp = 20;	begin	r1, 0, dac1, 0);	//motion	ses % 2;	}
int tempValue =	automatically	// OLULA al. anal			
int tempvalue =	•	// GUItool: end			
0;	generated code	automatically	Serial.begin(960	Serial.println(Bc	
0; int	generated code AudioSynthWav		0);	ount);	
0; int tempReading =	generated code AudioSynthWav eform	automatically	- '	ount); triggerState =	//write photocell
0; int tempReading = 0;	generated code AudioSynthWav eform waveform1; /	automatically generated code	0); imu.begin();	ount);	//write photocell if (Bcount == 1){
0; int tempReading = 0; int tempFreq =	generated code AudioSynthWav eform waveform1; / /xy=102,348	automatically generated code // GUltool: end	0); imu.begin(); filter.begin(100);	ount); triggerState = Bstate;	if (Bcount == 1){
0; int tempReading = 0;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav	automatically generated code  // GUltool: end automatically	0); imu.begin();	ount); triggerState = Bstate; if (Bcount ==	if (Bcount == 1){ int Pval =
0; int tempReading = 0; int tempFreq = 0;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform	automatically generated code // GUltool: end	0); imu.begin(); filter.begin(100);	ount); triggerState = Bstate;	if (Bcount == 1){  int Pval =  analogRead(2) /
0; int tempReading = 0; int tempFreq = 0; //motionf	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; /	automatically generated code  // GUltool: end automatically generated code	0); imu.begin(); filter.begin(100); }	ount); triggerState = Bstate; if (Bcount == 0){	if (Bcount == 1){ int Pval =
0; int tempReading = 0; int tempFreq = 0; //motionf int rollFreq = 0;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405	automatically generated code  // GUltool: end automatically generated code  #include	0); imu.begin(); filter.begin(100); } void loop()	ount); triggerState = Bstate; if (Bcount == 0){ digitalWrite(Led,	if (Bcount == 1){  int Pval =  analogRead(2) /  4;
0; int tempReading = 0; int tempFreq = 0; //motionf int rollFreq = 0; int pitchFreq =	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse< td=""><td>0); imu.begin(); filter.begin(100); }</td><td>ount); triggerState = Bstate; if (Bcount == 0){</td><td><pre>if (Bcount == 1){ int Pval = analogRead(2) / 4; //</pre></td></nxpmotionse<>	0); imu.begin(); filter.begin(100); }	ount); triggerState = Bstate; if (Bcount == 0){	<pre>if (Bcount == 1){ int Pval = analogRead(2) / 4; //</pre>
0; int tempReading = 0; int tempFreq = 0; //motionf int rollFreq = 0;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405	automatically generated code  // GUltool: end automatically generated code  #include	0); imu.begin(); filter.begin(100); } void loop()	ount); triggerState = Bstate; if (Bcount == 0){ digitalWrite(Led,	if (Bcount == 1){  int Pval =  analogRead(2) /  4;
0; int tempReading = 0; int tempFreq = 0; //motionf int rollFreq = 0; int pitchFreq = 0;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""></nxpmotionse>	0); imu.begin(); filter.begin(100); } void loop() {	<pre>ount);   triggerState = Bstate;  if (Bcount == 0){  digitalWrite(Led, LOW); }</pre>	<pre>if (Bcount == 1){ int Pval = analogRead(2) / 4; // EEPROM.write(</pre>
0; int tempReading = 0; int tempFreq = 0; //motionf int rollFreq = 0; int pitchFreq = 0;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu lated sine_fm1;	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""> #include  <wire.h> #include  #include</wire.h></nxpmotionse>	0); imu.begin();  filter.begin(100); }  void loop() { //amplitude amReading =	<pre>ount);   triggerState = Bstate;  if (Bcount == 0){  digitalWrite(Led, LOW);   }   if(Bcount == 1){</pre>	<pre>if (Bcount == 1){ int Pval = analogRead(2) / 4;  // EEPROM.write( addr,Pval); //addr = addr + 1;</pre>
0; int tempReading = 0; int tempFreq = 0; //motionf int rollFreq = 0; int pitchFreq = 0;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu lated sine_fm1; //xy=147,527	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""> #include  <wire.h></wire.h></nxpmotionse>	0); imu.begin();  filter.begin(100); }  void loop() { //amplitude	<pre>ount);   triggerState = Bstate;  if (Bcount == 0){  digitalWrite(Led, LOW);   }   if(Bcount ==</pre>	<pre>if (Bcount == 1){ int Pval = analogRead(2) / 4;  // EEPROM.write( addr,Pval); //addr = addr +</pre>
0; int tempReading = 0; int tempFreq = 0; //motionf int rollFreq = 0; int pitchFreq = 0;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu lated sine_fm1; //xy=147,527 AudioSynthWav eform waveform4; /	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""> #include  <wire.h> #include  #include</wire.h></nxpmotionse>	0); imu.begin();  filter.begin(100); }  void loop() { //amplitude amReading =	<pre>ount);   triggerState = Bstate;  if (Bcount == 0){  digitalWrite(Led, LOW);   }   if(Bcount == 1){</pre>	<pre>if (Bcount == 1){ int Pval = analogRead(2) / 4;  // EEPROM.write( addr,Pval); //addr = addr + 1;</pre>
0; int tempReading = 0; int tempFreq = 0;  //motionf int rollFreq = 0; int pitchFreq = 0; int yawFreq = 0; int amPot = 22; float	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu lated sine_fm1; //xy=147,527 AudioSynthWav eform waveform4; / /xy=148,603	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""> #include <wire.h> #include <eeprom.h>  NXPMotionSen</eeprom.h></wire.h></nxpmotionse>	0); imu.begin();  filter.begin(100); }  void loop() {  //amplitude  amReading =  analogRead(am  Pot);  level =	<pre>ount);   triggerState = Bstate;  if (Bcount == 0){  digitalWrite(Led, LOW);   }   if(Bcount == 1){  digitalWrite(Led, HIGH); }</pre>	<pre>if (Bcount == 1){ int Pval = analogRead(2) / 4;  // EEPROM.write( addr,Pval); //addr = addr + 1; // if(addr == EEPROM.length 0)</pre>
0; int tempReading = 0; int tempFreq = 0;  //motionf int rollFreq = 0; int pitchFreq = 0; int yawFreq = 0; int amPot = 22; float amReading = 0;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu lated sine_fm1; //xy=147,527 AudioSynthWav eform waveform4; / /xy=148,603 AudioMixer4	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""> #include <wire.h> #include <eeprom.h>  NXPMotionSen se imu;</eeprom.h></wire.h></nxpmotionse>	0); imu.begin();  filter.begin(100); }  void loop() {  //amplitude  amReading = analogRead(am Pot); level = amReading/	<pre>ount);   triggerState = Bstate;    if (Bcount == 0){    digitalWrite(Led,   LOW);    }   if(Bcount == 1){    digitalWrite(Led,   HIGH);    } }else{</pre>	<pre>if (Bcount == 1){ int Pval = analogRead(2) / 4;  // EEPROM.write( addr,Pval); //addr = addr + 1; // if(addr == EEPROM.length 0) // addr = 0;</pre>
0; int tempReading = 0; int tempFreq = 0;  //motionf int rollFreq = 0; int pitchFreq = 0; int yawFreq = 0; int amPot = 22; float	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu lated sine_fm1; //xy=147,527 AudioSynthWav eform waveform4; / /xy=148,603 AudioMixer4 mixer1; //	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""> #include <wire.h> #include <eeprom.h>  NXPMotionSen se imu; NXPSensorFusi</eeprom.h></wire.h></nxpmotionse>	0); imu.begin();  filter.begin(100); }  void loop() {  //amplitude  amReading =  analogRead(am  Pot);  level =	<pre>ount);   triggerState = Bstate;    if (Bcount == 0){    digitalWrite(Led,   LOW);    }   if(Bcount == 1){    digitalWrite(Led,   HIGH);    } }else{   triggerState =</pre>	<pre>if (Bcount == 1){  int Pval =   analogRead(2) /   4;  // EEPROM.write(   addr,Pval);   //addr = addr +   1;   // if(addr ==   EEPROM.length   0)   // addr = 0;   //</pre>
0; int tempReading = 0; int tempFreq = 0;  //motionf int rollFreq = 0; int pitchFreq = 0; int yawFreq = 0; int amPot = 22; float amReading = 0;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu lated sine_fm1; //xy=147,527 AudioSynthWav eform waveform4; / /xy=148,603 AudioMixer4 mixer1; // xy=295,391	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""> #include <wire.h> #include <eeprom.h>  NXPMotionSen se imu;</eeprom.h></wire.h></nxpmotionse>	0); imu.begin();  filter.begin(100); }  void loop() {  //amplitude  amReading = analogRead(am Pot); level = amReading/ 4096;	<pre>ount);   triggerState = Bstate;  if (Bcount == 0){  digitalWrite(Led, LOW);   }   if(Bcount == 1){  digitalWrite(Led, HIGH);   } }else{   triggerState = Bstate;</pre>	<pre>if (Bcount == 1){  int Pval =     analogRead(2) /     4;  //  EEPROM.write(     addr,Pval); //addr = addr +     1; // if(addr ==     EEPROM.length     0) // addr = 0; // // delay(200);</pre>
0; int tempReading = 0; int tempFreq = 0;  //motionf int rollFreq = 0; int pitchFreq = 0; int yawFreq = 0; int amPot = 22; float amReading = 0; float level = 0;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu lated sine_fm1; //xy=147,527 AudioSynthWav eform waveform4; / /xy=148,603 AudioMixer4 mixer1; // xy=295,391 AudioSynthWav	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""> #include <wire.h> #include <eeprom.h>  NXPMotionSen se imu; NXPSensorFusi</eeprom.h></wire.h></nxpmotionse>	0); imu.begin();  filter.begin(100); }  void loop() {  //amplitude  amReading = analogRead(am Pot); level = amReading/	<pre>ount);   triggerState = Bstate;    if (Bcount == 0){    digitalWrite(Led,   LOW);    }   if(Bcount == 1){    digitalWrite(Led,   HIGH);    }   }else{     triggerState =   Bstate;   }</pre>	<pre>if (Bcount == 1){  int Pval =   analogRead(2) /   4;  // EEPROM.write(   addr,Pval);   //addr = addr +   1;   // if(addr ==   EEPROM.length   0)   // addr = 0;   //</pre>
0; int tempReading = 0; int tempFreq = 0;  //motionf int rollFreq = 0; int pitchFreq = 0; int yawFreq = 0; int amPot = 22; float amReading = 0; float level = 0;  unsigned int	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu lated sine_fm1; //xy=147,527 AudioSynthWav eform waveform4; / /xy=148,603 AudioMixer4 mixer1; // xy=295,391 AudioSynthWav eform	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""> #include <wire.h> #include <eeprom.h>  NXPMotionSen se imu; NXPSensorFusi on filter;</eeprom.h></wire.h></nxpmotionse>	0); imu.begin();  filter.begin(100); }  void loop() {  //amplitude  amReading = analogRead(am Pot); level = amReading/ 4096;  //waveforms	<pre>ount);   triggerState = Bstate;  if (Bcount == 0){  digitalWrite(Led, LOW);   }   if(Bcount == 1){  digitalWrite(Led, HIGH);   } }else{   triggerState = Bstate;</pre>	<pre>if (Bcount == 1){  int Pval =     analogRead(2) /     4;  //  EEPROM.write(     addr,Pval); //addr = addr +     1; // if(addr ==     EEPROM.length     0) // addr = 0; // // delay(200); //}</pre>
0; int tempReading = 0; int tempFreq = 0;  //motionf int rollFreq = 0; int pitchFreq = 0; int yawFreq = 0; int amPot = 22; float amReading = 0; float level = 0;  unsigned int addr = 0;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu lated sine_fm1; //xy=147,527 AudioSynthWav eform waveform4; / /xy=148,603 AudioMixer4 mixer1; // xy=295,391 AudioSynthWav eform waveform waveform3; /	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""> #include <wire.h> #include <eeprom.h>  NXPMotionSen se imu; NXPSensorFusi</eeprom.h></wire.h></nxpmotionse>	0); imu.begin();  filter.begin(100); }  void loop() { //amplitude  amReading = analogRead(am Pot); level = amReading/ 4096; //waveforms  waveform1.freq	<pre>ount);   triggerState = Bstate;    if (Bcount == 0){    digitalWrite(Led,    LOW);    }   if(Bcount == 1){    digitalWrite(Led,   HIGH);    } }else{    triggerState = Bstate; } </pre>	<pre>if (Bcount == 1){  int Pval =     analogRead(2) /     4;  //  EEPROM.write(     addr,Pval); //addr = addr +     1; // if(addr ==     EEPROM.length     0) // addr = 0; // // delay(200); //}  //readings[100]</pre>
0; int tempReading = 0; int tempFreq = 0;  //motionf int rollFreq = 0; int pitchFreq = 0; int yawFreq = 0; int amPot = 22; float amReading = 0; float level = 0;  unsigned int addr = 0; int address = 0;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu lated sine_fm1; //xy=147,527 AudioSynthWav eform waveform4; / /xy=148,603 AudioMixer4 mixer1; // xy=295,391 AudioSynthWav eform waveform3; / /xy=300,517	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""> #include <wire.h> #include <eeprom.h>  NXPMotionSen se imu; NXPSensorFusi on filter;</eeprom.h></wire.h></nxpmotionse>	0); imu.begin();  filter.begin(100); }  void loop() {  //amplitude  amReading = analogRead(am Pot); level = amReading/4096;  //waveforms  waveform1.freq uency(Pfreq);	<pre>ount);   triggerState = Bstate;    if (Bcount == 0){    digitalWrite(Led,   LOW);    }   if(Bcount == 1){    digitalWrite(Led,   HIGH);    } }else{    triggerState = Bstate; }  // play toggle</pre>	<pre>if (Bcount == 1){  int Pval =   analogRead(2) /   4;  // EEPROM.write(   addr,Pval); //addr = addr +   1; // if(addr ==   EEPROM.length   0) // addr = 0; // // delay(200); //}  //readings[100]   = Pval;</pre>
0; int tempReading = 0; int tempFreq = 0;  //motionf int rollFreq = 0; int pitchFreq = 0; int yawFreq = 0; int amPot = 22; float amReading = 0; float level = 0;  unsigned int addr = 0;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu lated sine_fm1; //xy=147,527 AudioSynthWav eform waveform4; / /xy=148,603 AudioMixer4 mixer1; // xy=295,391 AudioSynthWav eform waveform3; / /xy=300,517 AudioFilterState	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""> #include <wire.h> #include <eeprom.h>  NXPMotionSen se imu; NXPSensorFusi on filter;</eeprom.h></wire.h></nxpmotionse>	0); imu.begin();  filter.begin(100); }  void loop() {  //amplitude  amReading = analogRead(am Pot); level = amReading/4096;  //waveforms  waveform1.freq uency(Pfreq); waveform1.amp	<pre>ount);   triggerState = Bstate;    if (Bcount == 0){    digitalWrite(Led,   LOW);    }   if(Bcount == 1){    digitalWrite(Led,   HIGH);    } }else{    triggerState = Bstate; }  // play toggle Pstate =</pre>	<pre>if (Bcount == 1){  int Pval =   analogRead(2) /   4;  // EEPROM.write(   addr,Pval); //addr = addr +   1; // if(addr ==   EEPROM.length   0) // addr = 0; // // delay(200); //}  //readings[100]   = Pval; //readings[100]</pre>
0; int tempReading = 0; int tempFreq = 0;  //motionf int rollFreq = 0; int pitchFreq = 0; int yawFreq = 0; int amPot = 22; float amReading = 0; float level = 0;  unsigned int addr = 0; int address = 0;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu lated sine_fm1; //xy=147,527 AudioSynthWav eform waveform4; / /xy=148,603 AudioMixer4 mixer1; // xy=295,391 AudioSynthWav eform waveform3; / /xy=300,517 AudioFilterState Variable filter1;	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""> #include <wire.h> #include <eeprom.h>  NXPMotionSen se imu; NXPSensorFusi on filter;  void setup(){</eeprom.h></wire.h></nxpmotionse>	0); imu.begin();  filter.begin(100); }  void loop() {  //amplitude  amReading = analogRead(am Pot); level = amReading/4096;  //waveforms  waveform1.freq uency(Pfreq); waveform1.amp litude(level);	<pre>ount);   triggerState = Bstate;    if (Bcount == 0){    digitalWrite(Led,   LOW);    }   if(Bcount == 1){    digitalWrite(Led,   HIGH);    } }else{    triggerState =   Bstate; }  // play toggle   Pstate =   digitalRead(Pbu</pre>	if (Bcount == 1){  int Pval =   analogRead(2) /   4;   //  EEPROM.write(   addr,Pval);  //addr = addr +   1;  // if(addr ==   EEPROM.length   0)  // addr = 0;  // delay(200);  //}  //readings[100]   = Pval;  //readings[100]   = map(Pval, 1,
0; int tempReading = 0; int tempFreq = 0;  //motionf int rollFreq = 0; int pitchFreq = 0; int yawFreq = 0; int amPot = 22; float amReading = 0; float level = 0;  unsigned int addr = 0; int address = 0;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu lated sine_fm1; //xy=147,527 AudioSynthWav eform waveform4; / /xy=148,603 AudioMixer4 mixer1; // xy=295,391 AudioSynthWav eform waveform3; / xy=295,391 AudioSynthWav eform waveform3; / /xy=300,517 AudioFilterState Variable filter1; //xy=432,393	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""> #include <wire.h> #include <eeprom.h>  NXPMotionSen se imu; NXPSensorFusi on filter;  void setup(){</eeprom.h></wire.h></nxpmotionse>	0); imu.begin();  filter.begin(100); }  void loop() {  //amplitude  amReading = analogRead(am Pot); level = amReading/ 4096;  //waveforms  waveform1.freq uency(Pfreq); waveform1.amp litude(level); waveform2.freq	<pre>ount);   triggerState = Bstate;    if (Bcount == 0){    digitalWrite(Led,   LOW);    }   if(Bcount == 1){    digitalWrite(Led,   HIGH);    } }else{    triggerState = Bstate; }  // play toggle Pstate =</pre>	if (Bcount == 1){  int Pval =     analogRead(2) /     4;  //  EEPROM.write(     addr,Pval);  //addr = addr +     1;  // if(addr ==     EEPROM.length     0)  // addr = 0;  // delay(200);  //}  //readings[100]     = Pval;  //readings[100]     = map(Pval, 1,     1023, 1, 500);
0; int tempReading = 0; int tempFreq = 0;  //motionf int rollFreq = 0; int pitchFreq = 0; int yawFreq = 0; int amPot = 22; float amReading = 0; float level = 0;  unsigned int addr = 0; int address = 0; byte value;	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu lated sine_fm1; //xy=147,527 AudioSynthWav eform waveform4; / /xy=148,603 AudioMixer4 mixer1; // xy=295,391 AudioSynthWav eform waveform3; / /xy=300,517 AudioFilterState Variable filter1;	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""> #include <wire.h> #include <eeprom.h>  NXPMotionSen se imu; NXPSensorFusi on filter;  void setup(){</eeprom.h></wire.h></nxpmotionse>	0); imu.begin();  filter.begin(100); }  void loop() {  //amplitude  amReading = analogRead(am Pot); level = amReading/4096;  //waveforms  waveform1.freq uency(Pfreq); waveform1.amp litude(level);	<pre>ount);   triggerState = Bstate;    if (Bcount == 0){    digitalWrite(Led,   LOW);    }   if(Bcount == 1){    digitalWrite(Led,   HIGH);    } }else{    triggerState =   Bstate; }  // play toggle   Pstate =   digitalRead(Pbu</pre>	if (Bcount == 1){  int Pval =   analogRead(2) /   4;   //  EEPROM.write(   addr,Pval);  //addr = addr +   1;  // if(addr ==   EEPROM.length   0)  // addr = 0;  // delay(200);  //}  //readings[100]   = Pval;  //readings[100]   = map(Pval, 1,
0; int tempReading = 0; int tempFreq = 0;  //motionf int rollFreq = 0; int pitchFreq = 0; int yawFreq = 0; int amPot = 22; float amReading = 0; float level = 0;  unsigned int addr = 0; int address = 0; byte value;  #include	generated code AudioSynthWav eform waveform1; / /xy=102,348 AudioSynthWav eform waveform2; / /xy=105,405 AudioSynthWav eformSineModu lated sine_fm1; //xy=147,527 AudioSynthWav eform waveform4; / /xy=148,603 AudioMixer4 mixer1; // xy=295,391 AudioSynthWav eform waveform3; / xy=295,391 AudioSynthWav eform waveform3; / /xy=300,517 AudioFilterState Variable filter1; //xy=432,393 AudioOutputAn	automatically generated code  // GUltool: end automatically generated code  #include <nxpmotionse nse.h=""> #include <wire.h> #include <eeprom.h>  NXPMotionSen se imu; NXPSensorFusi on filter;  void setup(){</eeprom.h></wire.h></nxpmotionse>	0); imu.begin();  filter.begin(100); }  void loop() {  //amplitude  amReading = analogRead(am Pot); level = amReading/ 4096;  //waveforms  waveform1.freq uency(Pfreq); waveform1.amp litude(level); waveform2.freq	ount); triggerState = Bstate;  if (Bcount == 0){  digitalWrite(Led, LOW); } if(Bcount == 1){  digitalWrite(Led, HIGH); } }else{ triggerState = Bstate; }  // play toggle Pstate = digitalRead(Pbu tton);	if (Bcount == 1){  int Pval =     analogRead(2) /     4;  //  EEPROM.write(     addr,Pval);  //addr = addr +     1;  // if(addr ==     EEPROM.length     0)  // addr = 0;  // delay(200);  //}  //readings[100]     = Pval;  //readings[100]     = map(Pval, 1,     1023, 1, 500);  //addr = addr +

//if(addr ==	heading =	//	//	}	else if (cfg ==
readings[100])	filter.getYaw();	Serial.println("N	Serial.println("	// Serial.print("	0x40) raw = raw
// addr = 0;	//	o more	Chip =	CRC=");	& ~1; // 11 bit
Pfreq =	Serial.print("Ori	addresses.");	DS1822");	//	res, 375 ms
map(Pval, 1,	entation: ");	//	$type\_s = 0;$	Serial.print(One	//// default is
1023, 1, 1000);	//	Serial.println();	break;	Wire::crc8(data,	12 bit
, , , ,	Serial.print(head	I- V/	default:	8), HEX);	resolution, 750
	ing);	ds.reset_search	//	//	ms conversion
// Serial.print	// Serial.print("	0;	Serial.println("D	Serial.println();	time
(Pval);	");	delay(250);	evice is not a	1 0,	}
//	//	return;	DS18x20 family	// Convert the	celsius =
waveform1.freq	Serial.print(pitch	}	device.");	data to actual	(float)raw / 16.0;
uency(Pfreq);	);	,	return;	temperature	fahrenheit =
//	// Serial.print("	//	}	// because the	celsius * 1.8 +
waveform1.amp	");	Serial.print("RO	·	result is a 16 bit	32.0;
litude(level);	<i>,</i> //	M =");	ds.reset();	signed integer,	// Serial.print("
	Serial.println(roll	for $(i = 0; i < 8;$		it should	Temperature =
delay(10);	);	i++) {	ds.select(addr);	// be stored to	");
}		// Serial.write(	ds.write(0x44,	an "int16_t"	//
	rollFreq =	');	1); // start	type, which is	Serial.print(celsi
	map(roll, -180,	//	conversion, with	always 16 bits	us);
	180, 1, 500);	Serial.print(addr	parasite power	// even when	// Serial.print("
//motion	pitchFreq =	[i], HEX);	on at the end	compiled on a	Celsius, ");
	map(pitch,	}		32 bit	//
if (Bcount == 1){	-180, 180, 1,		delay(10); //	processor.	Serial.print(fahre
	500);	if	maybe 750ms	int16_t raw =	nheit);
float ax, ay, az;	yawFreq =	(OneWire::crc8(	is enough,	(data[1] << 8)	//
float gx, gy,	map(heading,	addr, 7) !=	maybe not	data[0];	Serial.println("
gz;	-180, 180, 1,	addr[7]) {	// we might do	if (type_s) {	Fahrenheit");
float mx, my,	500);	//	a ds.depower()	raw = raw <<	
mz;		Serial.println("C	here, but the	3; // 9 bit	//fahrenheit =
float roll, pitch,	//	RC is not	reset will take	resolution	tempReading;
heading;	waveform2.freq	valid!");	care of it.	default	tempFreq =
:4	uency(Rollfreq);	return;		if (data[7] ==	map(fahrenheit,
if	//	}	present =	0x10) {	-67, 257, 1,
(imu.available())	waveform2.amp	// Carial println():	ds.reset();	// "count	1023); //tamp\/alua-
۱ // Read the	litude(level);	Serial.println();	ds.select(addr);	remain" gives full 12 bit	//tempValue= map(tempReadi
motion sensors	delay(10);	// the first	ds.write(0xBE);	resolution	ng, -67., 257.,
111011011 3013013	} }	ROM byte	// Read	raw = (raw &	1, 1023.);
imu.readMotion	J	indicates which	Scratchpad	0xFFF0) + 12 -	//tempFreq =
Sensor(ax, ay,		chip	Ociatoripad	data[6];	map(tempValue,
az, gx, gy, gz,	//temp	switch	// Serial.print("	ααια <sub>[</sub> ο], }	1, 1023, 1, 500);
mx, my, mz);	// tomp	(addr[0]) {	Data = ");	} else {	//
11DX, 111 <b>y</b> , 111 <b>2</b> /,	if (Bcount == 1){	case 0x10:	//	byte cfg =	waveform3.freq
// Update the	(2000)(	//	Serial.print(pres	(data[4] & 0x60);	uency(tempFreq
SensorFusion		Serial.println("	ent, HEX);	// at lower	);
filter	byte i;	Chip =	// Serial.print("	res, the low bits	,, //
	byte present =	DS18S20"); //	");	are undefined,	waveform3.amp
filter.update(gx,	0;	or old DS1820	for $(i = 0; i < 9;$	so let's zero	litude(level);
gy, gz, ax, ay,	byte type_s;	$type_s = 1;$	i++) { //	them	//Serial.print
az, mx, my, mz);	byte data[12];	break;	we need 9	if (cfg ==	(tempFreq);
//	byte addr[8];	case 0x28:	bytes	0x00) raw = raw	//delay(10);
// // print the	float celsius,	//	data[i] =	& ~7; // 9 bit	}
heading, pitch	fahrenheit;	Serial.println("	ds.read();	resolution,	}
and roll		Chip =	//	93.75 ms	
roll =	if (!	DS18B20");	Serial.print(data[	else if (cfg ==	
filter.getRoll();	ds.search(addr))	$type\_s = 0;$	i], HEX);	0x20) raw = raw	
pitch =	{	break;	// Serial.print("	& ~3; // 10 bit	
filter.getPitch();		case 0x22:	");	res, 187.5 ms	

## **Environmental Composition**

Keanu yokoyama

Where you are right now, Start recording.

For the next thirty seconds, think of a place to visit. Stop recording.

Go to that place, and for the last thirty seconds of your journey, Start recording.

Arrive at your destination. Stop recording.

Find an area you find you find interesting, and for thirty seconds, Start recording.

Observe your surrounds, Stop recording.

Explore whats around you, and take a thirty second walk, Start recording.

When you are finished, Stop recording.