**Project -01**

//Arduino code:  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
int val ; //define val  
int ledpin=13; // define of digi tal interface 13  
void setup()  
{S  
erial .begin (9600);   
// set the baud rate is 9600.  
pinMode (ledpin, OUTPUT);   
// set 13 for the output port .  
}v  
oid loop()  
{v  
al=Serial.read();//   
//read the PC machine send instructionsor   
//character of Arduino, and the instruction  
//or character is assigned to val  
if (val=='R' ) // to judge whether  
//the instruction or character received is "R".  
{/  
/ if recei ved "R" character  
digitalWrite (ledpin, HIGH);  
// digi tal 13 LED on  
delay (500);  
digitalWrite (ledpin, LOW);   
// digi tal 13 LED off  
delay (500);  
Serial .println ("HelloWorld!");  
// display "Hel lo World!"  
}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -02**

/\*  
Blink  
Turns on an LED on for one second, then off for one second,  
repeatedly.  
This example code is in the public domain.  
\*/  
// Pin 13 has an LED connected on most Arduino boards.  
// give it a name:  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
int led = 13;  
// the setup routine runs once when you press reset:  
void setup()   
{   
// initialize the digital pin as an output.  
pinMode(led, OUTPUT);   
} /  
/ the loop routine runs over and over again forever:  
void loop()  
{  
digitalWrite(led, HIGH);   
// turn the LED on (HIGH is the voltage level)  
delay(1000); // wait for a second  
digitalWrite(led, LOW);   
// turn the LED off by making the voltage LOW  
delay(1000); // wait for a second  
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -03**

//arduino code :  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
int redled =10; // define digital interface 10  
int yellowled =7; // define digital interface 7  
int greenled =4; // define digital interface 4  
void setup ()  
{p  
inMode (redled, OUTPUT); //define red light for the output  
pinMode (yellowled, OUTPUT);//define yellow light for output  
pinMode (greenled, OUTPUT); //define green light for output  
}  
void loop ()  
{d  
igitalWrite (redled, HIGH); // red light on  
delay (1000); // 1 second delay  
digitalWrite (redled, LOW); // red light off  
digitalWrite (yellowled, HIGH); // yellow light on  
delay (200); // 0.2 second delay  
digitalWrite (yellowled, LOW); // yellow light off  
digitalWrite (greenled, HIGH); // green light on  
delay (1000); // 1 second delay  
digitalWrite (greenled, LOW); // green light off  
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -04**

This example code  
/\*For Loop Iteration \*/  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
int timer = 100; // The higher the number, the slower the timing.  
void setup()  
{  
// use a for loop to initialize each pin as an output:  
for (int thisPin = 2; thisPin < 8; thisPin++) {  
pinMode(thisPin, OUTPUT);   
}  
} v  
oid loop()   
{  
// loop from the lowest pin to the highest:  
for (int thisPin = 2; thisPin < 8; thisPin++) {   
// turn the pin on:  
digitalWrite(thisPin, HIGH);   
delay(timer);   
// turn the pin off:  
digitalWrite(thisPin, LOW);   
}  
// loop from the highest pin to the lowest:  
for (int thisPin = 7; thisPin >= 2; thisPin--) {   
// turn the pin on:  
digitalWrite(thisPin, HIGH);  
delay(timer);  
// turn the pin off:  
digitalWrite(thisPin, LOW);  
}  
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -05**

Arduino Code  
/\*The example shows how to fade an LED using the  
analogWrite \*  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
int ledPin = 9; // LED connected to digital pin 9  
void setup() {   
// nothing happens in setup   
}   
void loop() {   
// fade in from min to max in increments of 5 points:  
for(int fadeValue = 0 ; fadeValue <= 255; fadeValue +=5)   
{   
// sets the value (range from 0 to 255):  
analogWrite(ledPin, fadeValue);   
// wait for 30 milliseconds to see the dimming effect   
delay(30);   
}   
// fade out from max to min in increments of 5 points:  
for(int fadeValue = 255 ; fadeValue >= 0; fadeValue -=5) {   
// sets the value (range from 0 to 255):  
analogWrite(ledPin, fadeValue);   
// wait for 30 milliseconds to see the dimming effect   
delay(30);   
}   
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -06**

Arduino Code  
/\* Button\*/  
// constants won't change. They're used here to   
// set pin numbers:  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
const int buttonPin = 2; // the number of the pushbutton pin  
const int ledPin = 13; // the number of the LED pin  
// variables will change:  
int buttonState = 0; // variable for reading the pushbutton status  
void setup() {  
// initialize the LED pin as an output:  
pinMode(ledPin, OUTPUT);   
// initialize the pushbutton pin as an input:  
pinMode(buttonPin, INPUT);   
} v  
oid loop()  
{  
// read the state of the pushbutton value:  
buttonState = digitalRead(buttonPin);  
// check if the pushbutton is pressed.  
// if it is, the buttonState is HIGH:  
if (buttonState == HIGH) {   
// turn LED on:   
digitalWrite(ledPin, HIGH);   
}   
else {  
// turn LED off:  
digitalWrite(ledPin, LOW);   
}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -07**

//Ardui no code:  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
int redled=10;  
int yellowled=9;  
int greenled=8;  
int redpin=7;  
int yellowpin=6;  
int greenpin=5;  
int red;  
int yellow;  
int green;  
void setup ()  
{p  
inMode (redled, OUTPUT);  
pinMode (yellowled, OUTPUT);  
pinMode (greenled, OUTPUT);  
pinMode (redpin, INPUT);  
pinMode (yellowpin, INPUT);  
pinMode (greenpin, INPUT);  
}v  
oid loop ()  
{r  
ed=digitalRead (redpin);  
if ( red==LOW)  
{digitalWrite (redled, LOW); }  
else  
{digitalWrite (redled, HIGH); }  
yellow=digitalRead (yellowpin);  
if ( yellow==LOW)  
{digitalWrite (yellowled, LOW); }  
else  
{digitalWrite (yellowled, HIGH); }  
green=digitalRead (greenpin);  
if ( green==LOW)  
{digitalWrite (greenled, LOW); }  
else  
{digitalWrite (greenled, HIGH); }  
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -08**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*int red = 11; //this sets the red led pin  
int green = 10; //this sets the green led pin  
int blue = 9; //this sets the blue led pin  
int redNow;  
int blueNow;  
int greenNow;  
int redNew;  
int blueNew;  
int greenNew;  
void setup()  
{ //this sets the output pins  
pinMode(red, OUTPUT);  
pinMode(green, OUTPUT);  
pinMode(blue, OUTPUT);  
redNow = random(255);  
blueNow = random(255);  
greenNow = random(255);  
redNew = redNow;  
blueNew = blueNow;  
greenNew = greenNow;  
} #  
define fade(x,y) if (x>y) x--; else if (x<y) x++;  
void loop()  
{a  
nalogWrite(blue, blueNow);  
analogWrite(red, redNow);  
analogWrite(green, greenNow);  
redNew = random(255);  
blueNew = random(255);  
greenNew = random(255);  
// fade to new colors  
while ((redNow != redNew) ||  
(blueNow != blueNew) ||  
(greenNow != greenNew))  
{  
fade(redNow,redNew)  
fade(blueNow,blueNew)  
fade(greenNow,greenNew)  
analogWrite(blue, blueNow);  
analogWrite(red, redNow);  
analogWrite(green, greenNow);  
delay(20);  
}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -09**

Arduino Code  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
int speakerPin = 9;  
int length = 15; // the number of notes  
char notes[] = "ccggaagffeeddc "; // a space represents a rest  
int beats[] = { 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 2, 4 };  
int tempo = 300;  
void playTone(int tone, int duration) {  
for (long i = 0; i < duration \* 1000L; i += tone \* 2) {  
digitalWrite(speakerPin, HIGH);  
delayMicroseconds(tone);  
digitalWrite(speakerPin, LOW);  
delayMicroseconds(tone);  
}  
} v  
oid playNote(char note, int duration) {  
char names[] = { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C' };  
int tones[] = { 1915, 1700, 1519, 1432, 1275, 1136, 1014, 956 };  
// play the tone corresponding to the note name  
for (int i = 0; i < 8; i++) {  
if (names[i] == note) {  
playTone(tones[i], duration);  
}  
}  
} v  
oid setup() {  
pinMode(speakerPin, OUTPUT);  
} v  
oid loop() {  
for (int i = 0; i < length; i++) {  
if (notes[i] == ' ') {  
delay(beats[i] \* tempo); // rest  
} else {  
playNote(notes[i], beats[i] \* tempo);  
}  
// pause between notes  
delay(tempo / 2);   
}  
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -10**

arduino code:  
/\*AnalogReadSerial  
Reads an analog input on pin 0,   
prints the result to the serial monitor.  
Attach the center pin of a potentiometer to pin A0,   
and the outside pins to +5V and ground. \*/  
// the setup routine runs once when you press reset:  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
void setup() {  
// initialize serial communication at 9600 bits per second:  
Serial.begin(9600);  
} /  
/ the loop routine runs over and over again forever:  
void loop() {  
// read the input on analog pin 0:  
int sensorValue = analogRead(A0);  
// print out the value you read:  
Serial.println(sensorValue);  
delay(1); // delay in between reads for stability  
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -11**

//Ardui no code:  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
void setup ()  
{p  
inMode (6, OUTPUT);  
}v  
oid loop ()  
{w  
hile (1)  
{c  
har i, j;  
while (1)  
{f  
or (i =0; i<80; i++  
)// a frequency voice  
{d  
igitalWrite (6, HIGH);  
delay (1);  
digitalWrite (6, LOW);  
delay (1);  
}f  
or (i =0; i<100; i++) // another frequency voice  
{d  
igitalWrite (6, HIGH);  
delay (2);  
digitalWrite (6, LOW);  
delay (2);  
}}}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -12**

//arduino code:  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
int potpin=A0; // define A0 is connected with the   
//photosensitive resistor  
int ledpin=11; // define 11 output PWM regulating   
// LED brightness  
int val=0; // define val  
void setup ()  
{p  
inMode (ledpin, OUTPUT); // define io 11 output   
Serial.begin (9600); // set the baud rate of 9600  
} v  
oid loop ()  
{ v  
al=analogRead (potpin); // analog read sensor   
//values are assigned to val  
Serial.println (val); // display the Val numerical variables  
analogWrite (ledpin, val); // open LED and set the brightness  
// (PWM output a maximum of 255)  
delay (10); // 0.01 second delay  
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -13**

Arduino code:  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*   
const int tiltPin = 2; // the number of the tilt pin  
const int ledPin = 13; // the number of the LED pin  
// variables will change:  
int tiltState = 0; // variable for reading the tilt status  
void setup()   
{  
// initialize the LED pin as an output:  
pinMode(ledPin, OUTPUT);   
// initialize the tilt pin as an input:  
pinMode(tiltPin, INPUT);   
} v  
oid loop()  
{  
// read the state of the tilt value:  
tiltState = digitalRead(tiltPin);  
// check if the tilt is Oblique.  
// if it is, the tiltState is HIGH:  
if (tiltState == HIGH) {   
// turn LED on:   
digitalWrite(ledPin, HIGH);   
}   
else {  
// turn LED off:  
digitalWrite(ledPin, LOW);   
}  
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -14**

Arduino code:  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
int flame=A5; // ANALOG A5 definition of flame  
int Beep=8; // define the buzzer for digital Io8  
int val=0; //DEFINE digital variables;  
void setup ()  
{p  
inMode (Beep, OUTPUT); // define buzzer output  
pinMode (flame, INPUT); // define flame sensor inpu  
tS  
erial.begin (9600); // set the baud rate to 9600  
}  
void loop ()  
{v  
al=analogRead (flame); // reading flame value  
Serial.println (val); // analog value, print it out  
if (val>=600) // when the analog value is big than 600   
{d  
igitalWrite (Beep, HIGH);  
}e  
lse  
{d  
igitalWrite (Beep, LOW);  
}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -15**

Arduino code:  
//lM35 Pin Variables  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
int sensorPin = 0;   
//the analog pin the lm35's Vout (sense) pin is connected to  
//the resolution is 10 mV / degree centigrade with a  
//500 mV offset to allow for negative temperatures  
/\*  
\* setup() - this function runs once when you turn your Arduino on  
\* We initialize the serial connection with the computer  
\*/  
void setup()  
{  
Serial.begin(9600); //Start the serial connection with the computer  
//to view the result open the serial monitor   
} v  
oid loop() // run over and over again  
{  
//getting the voltage reading from the temperature sensor  
int reading = analogRead(sensorPin);   
// converting that reading to voltage, for 3.3v arduino use 3.3  
float voltage = reading \* 5.0;  
voltage /= 1024.0;   
// print out the voltage  
Serial.print(voltage); Serial.println(" volts");  
// now print out the temperature  
float temperatureC = (voltage - 0.5) \* 100 ;   
//converting from 10 mv per degree wit 500 mV offset  
//to degrees ((voltage - 500mV) times 100)  
Serial.print(temperatureC); Serial.println(" degrees C");  
// now convert to Fahrenheit  
float temperatureF = (temperatureC \* 9.0 / 5.0) + 32.0;  
Serial.print(temperatureF); Serial.println(" degrees F");  
delay(1000);   
//waiting a second  
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -16**

Arduino code:  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
int data = 2;  
int clock = 4;  
int latch = 5;  
int ledState = 0;  
const int ON = HIGH;  
const int OFF = LOW;  
void setup()  
{p  
inMode(data, OUTPUT);  
pinMode(clock, OUTPUT);  
pinMode(latch, OUTPUT);  
}v  
oid loop()  
{i  
nt delayTime = 100;  
for(int i = 0; i < 256; i++)  
{u  
pdateLEDs(i);  
delay(delayTime);  
}}v  
oid updateLEDs(int value)  
{d  
igitalWrite(latch, LOW);  
shiftOut(data, clock, MSBFIRST, value);  
digitalWrite(latch, HIGH);  
}v  
oid updateLEDsLong(int value)  
{d  
igitalWrite(latch, LOW);  
for(int i = 0; i < 8; i++)  
{i  
nt bit = value & B10000000;  
value = value << 1;  
if(bit == 128){digitalWrite(data, HIGH);}  
else{digitalWrite(data, LOW);}  
digitalWrite(clock, HIGH);  
delay(1);  
digitalWrite(clock, LOW);  
}d  
igitalWrite(latch, HIGH);  
}i  
nt bits[]={B00000001, B00000010, B00000100,   
B00001000, B00010000, B00100000,  
B01000000, B10000000};  
int masks[] ={B11111110, B11111101, B11111011,  
B11110111, B11101111, B11011111,  
B10111111, B01111111};  
void changeLED(int led, int state)  
{l  
edState = ledState & masks[led];  
if(state == ON){ledState = ledState | bits[led];}  
updateLEDs(ledState);

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -17**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*int a = 7 ;// IO7 connect a

int b = 6 ;// IO66 connected to b

int c = 5 ;// IO5 connection c

int d = 11 ;// IO11 connection d

int e = 10 ;// IO10 connection e

int f = 8 ;// IO8 connection f

int g = 9 ;// IO9 connection g

int dp = 4 ;// IO4 connection dp

void digital\_1 (void) // display number 1

{u

nsigned char j;

digitalWrite (c, HIGH) ;// to the digital interface 5-pin high, light c

digitalWrite (b, HIGH) ;// lit b segment

for (j = 7; j <= 11; j++) // off remaining segments

digitalWrite (j, LOW);

digitalWrite (dp, LOW) ;// turn off decimal point DP segment

}v

oid digital\_2 (void) // display the number 2

{u

nsigned char j;

digitalWrite (b, HIGH);

digitalWrite (a, HIGH);

for (j = 9; j <= 11; j++)

digitalWrite (j, HIGH);

digitalWrite (dp, LOW);

digitalWrite (c, LOW);

digitalWrite (f, LOW);

}v

oid digital\_3 (void) // display the number 3

{u

nsigned char j;

digitalWrite (g, HIGH);

digitalWrite (d, LOW);

for (j = 5; j <= 7; j++)

digitalWrite (j, HIGH);

digitalWrite (dp, LOW);

digitalWrite (f, LOW);

digitalWrite (e, HIGH);

}v

oid digital\_4 (void) // display the number 4

{d

igitalWrite (c, HIGH);

digitalWrite (b, HIGH);

digitalWrite (f, HIGH);

digitalWrite (g, HIGH);

digitalWrite (dp, LOW);

digitalWrite (a, LOW);

digitalWrite (e, LOW);

digitalWrite (d, LOW);

}v

oid digital\_5 (void) // display the number 5

{u

nsigned char j;

for (j = 7; j <= 9; j++)

digitalWrite (j, HIGH);

digitalWrite (c, HIGH);

digitalWrite (d, LOW);

digitalWrite (dp, LOW);

digitalWrite (b, LOW);

digitalWrite (e, HIGH);

}v

oid digital\_6 (void) // display the number 6

{u

nsigned char j;

for (j = 7; j <= 11; j++)

digitalWrite (j, HIGH);

digitalWrite (c, HIGH);

digitalWrite (dp, LOW);

digitalWrite (b, LOW);

}v

oid digital\_7 (void) // display the number 7

{u

nsigned char j;

for (j = 5; j <= 7; j++)

digitalWrite (j, HIGH);

digitalWrite (dp, LOW);

for (j = 8; j <= 11; j++)

digitalWrite (j, LOW);

}v

oid digital\_8 (void) // display the number 8

{u

nsigned char j;

for (j = 5; j <= 11; j++)

digitalWrite (j, HIGH);

digitalWrite (dp, LOW);

}v

oid setup ()

{i

nt i ;// define variables

for (i = 4; i <= 11; i++)

pinMode (i, OUTPUT) ;// set 4 to 11 pin to output mode

}v

oid loop ()

{w

hile (1)

{d

igital\_1 () ;// display the number 1

delay (2000) ;// delay 2s

digital\_2 () ;// display the number 2

delay (1000); // delay 1s

digital\_3 () ;// display the number 3

delay (1000); // delay 1s

digital\_4 () ;// display the number 4

delay (1000); // delay 1s

digital\_5 () ;// display the number 5

delay (1000); // delay 1s

digital\_6 () ;// display the number 6

delay (1000); // delay 1s

digital\_7 () ;// display the number 7

delay (1000); // delay 1s

digital\_8 () ;// display the number 8

delay (1000); // delay 1s

}}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -18**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*int digit1 = 11; //PWM Display pin 1  
int digit2 = 10; //PWM Display pin 2  
int digit3 = 9; //PWM Display pin 6  
int digit4 = 6; //PWM Display pin 8  
int segA = A1; // Display pin 14  
int segB = 3; // Display pin 16  
int segC = 4; // Display pin 13  
int segD = 5; // Display pin 3  
int segE = A0; // Display pin 5  
int segF = 7; // Display pin 11  
int segG = 8; // Display pin 15  
void setup() {   
pinMode(segA, OUTPUT);  
pinMode(segB, OUTPUT);  
pinMode(segC, OUTPUT);  
pinMode(segD, OUTPUT);  
pinMode(segE, OUTPUT);  
pinMode(segF, OUTPUT);  
pinMode(segG, OUTPUT);  
pinMode(digit1, OUTPUT);  
pinMode(digit2, OUTPUT);  
pinMode(digit3, OUTPUT);  
pinMode(digit4, OUTPUT);  
pinMode(13, OUTPUT);  
}  
void loop() {  
// long startTime = millis();  
displayNumber(millis()/1000);  
// while( (millis() - startTime) < 2000) {  
// displayNumber(1217);  
// }  
// delay(1000);   
}  
// Given a number, we display 10:22  
// After running through the 4 numbers, the display   
is left turned off  
// Display brightness  
// Each digit is on for a certain amount of microseconds  
// Then it is off until we have reached a total of 20ms for the  
function call  
// Let's assume each digit is on for 1000us  
// If each digit is on for 1ms, there are 4 digits, so the display  
is off for 16ms.  
// That's a ratio of 1ms to 16ms or 6.25% on time (PWM).  
// Let's define a variable called brightness that varies from:  
// 5000 blindingly bright (15.7mA current draw per digit)  
// 2000 shockingly bright (11.4mA current draw per digit)  
// 1000 pretty bright (5.9mA)  
// 500 normal (3mA)  
// 200 dim but readable (1.4mA)  
// 50 dim but readable (0.56mA)  
// 5 dim but readable (0.31mA)  
// 1 dim but readable in dark (0.28mA)  
void displayNumber(int toDisplay) {  
#define DISPLAY\_BRIGHTNESS 500  
#define DIGIT\_ON LOW  
#define DIGIT\_OFF HIGH  
long beginTime = millis();  
for(int digit = 4 ; digit > 0 ; digit--) {  
// Turn on a digit for a short amount of time  
switch(digit) {  
case 1:  
digitalWrite(digit1, DIGIT\_ON);  
break;  
case 2:  
digitalWrite(digit2, DIGIT\_ON);  
break;  
case 3:  
digitalWrite(digit3, DIGIT\_ON);  
break;  
case 4:  
digitalWrite(digit4, DIGIT\_ON);  
break;  
}  
// Turn on the right segments for this digit  
lightNumber(toDisplay % 10);  
toDisplay /= 10;  
// Display this digit for a fraction of a second   
(between 1us and 5000us, 500 is pretty good)  
delayMicroseconds(DISPLAY\_BRIGHTNESS);   
// Turn off all segments  
lightNumber(10);   
// Turn off all digits  
digitalWrite(digit1, DIGIT\_OFF);  
digitalWrite(digit2, DIGIT\_OFF);  
digitalWrite(digit3, DIGIT\_OFF);  
digitalWrite(digit4, DIGIT\_OFF);  
}  
while( (millis() - beginTime) < 10) ;   
//Wait for 20ms to pass before  
we paint the display again  
}  
// Given a number, turns on those segments  
// If number == 10, then turn off number  
void lightNumber(int numberToDisplay) {  
#define SEGMENT\_ON HIGH  
#define SEGMENT\_OFF LOW  
switch (numberToDisplay){  
case 0:  
digitalWrite(segA, SEGMENT\_ON);  
digitalWrite(segB, SEGMENT\_ON);  
digitalWrite(segC, SEGMENT\_ON);  
digitalWrite(segD, SEGMENT\_ON);  
digitalWrite(segE, SEGMENT\_ON);  
digitalWrite(segF, SEGMENT\_ON);  
digitalWrite(segG, SEGMENT\_OFF);  
break;  
case 1:  
digitalWrite(segA, SEGMENT\_OFF);  
digitalWrite(segB, SEGMENT\_ON);  
digitalWrite(segC, SEGMENT\_ON);  
digitalWrite(segD, SEGMENT\_OFF);  
digitalWrite(segE, SEGMENT\_OFF);  
digitalWrite(segF, SEGMENT\_OFF);  
digitalWrite(segG, SEGMENT\_OFF);  
break;  
case 2:  
digitalWrite(segA, SEGMENT\_ON);  
digitalWrite(segB, SEGMENT\_ON);  
digitalWrite(segC, SEGMENT\_OFF);  
digitalWrite(segD, SEGMENT\_ON);  
digitalWrite(segE, SEGMENT\_ON);  
digitalWrite(segF, SEGMENT\_OFF);  
digitalWrite(segG, SEGMENT\_ON);  
break;  
case 3:  
digitalWrite(segA, SEGMENT\_ON);  
digitalWrite(segB, SEGMENT\_ON);  
digitalWrite(segC, SEGMENT\_ON);  
digitalWrite(segD, SEGMENT\_ON);  
digitalWrite(segE, SEGMENT\_OFF);  
digitalWrite(segF, SEGMENT\_OFF);  
digitalWrite(segG, SEGMENT\_ON);  
break;  
case 4:  
digitalWrite(segA, SEGMENT\_OFF);  
digitalWrite(segB, SEGMENT\_ON);  
digitalWrite(segC, SEGMENT\_ON);  
digitalWrite(segD, SEGMENT\_OFF);  
digitalWrite(segE, SEGMENT\_OFF);  
digitalWrite(segF, SEGMENT\_ON);  
digitalWrite(segG, SEGMENT\_ON);  
break;  
case 5:  
digitalWrite(segA, SEGMENT\_ON);  
digitalWrite(segB, SEGMENT\_OFF);  
digitalWrite(segC, SEGMENT\_ON);  
digitalWrite(segD, SEGMENT\_ON);  
digitalWrite(segE, SEGMENT\_OFF);  
digitalWrite(segF, SEGMENT\_ON);  
digitalWrite(segG, SEGMENT\_ON);  
break;  
case 6:  
digitalWrite(segA, SEGMENT\_ON);  
digitalWrite(segB, SEGMENT\_OFF);  
digitalWrite(segC, SEGMENT\_ON);  
digitalWrite(segD, SEGMENT\_ON);  
digitalWrite(segE, SEGMENT\_ON);  
digitalWrite(segF, SEGMENT\_ON);  
digitalWrite(segG, SEGMENT\_ON);  
break;  
case 7:  
digitalWrite(segA, SEGMENT\_ON);  
digitalWrite(segB, SEGMENT\_ON);  
digitalWrite(segC, SEGMENT\_ON);  
digitalWrite(segD, SEGMENT\_OFF);  
digitalWrite(segE, SEGMENT\_OFF);  
digitalWrite(segF, SEGMENT\_OFF);  
digitalWrite(segG, SEGMENT\_OFF);  
break;  
case 8:  
digitalWrite(segA, SEGMENT\_ON);  
digitalWrite(segB, SEGMENT\_ON);  
digitalWrite(segC, SEGMENT\_ON);  
digitalWrite(segD, SEGMENT\_ON);  
digitalWrite(segE, SEGMENT\_ON);  
digitalWrite(segF, SEGMENT\_ON);  
digitalWrite(segG, SEGMENT\_ON);  
break;  
case 9:  
digitalWrite(segA, SEGMENT\_ON);  
digitalWrite(segB, SEGMENT\_ON);  
digitalWrite(segC, SEGMENT\_ON);  
digitalWrite(segD, SEGMENT\_ON);  
digitalWrite(segE, SEGMENT\_OFF);  
digitalWrite(segF, SEGMENT\_ON);  
digitalWrite(segG, SEGMENT\_ON);  
break;  
case 10:  
digitalWrite(segA, SEGMENT\_OFF);  
digitalWrite(segB, SEGMENT\_OFF);  
digitalWrite(segC, SEGMENT\_OFF);  
digitalWrite(segD, SEGMENT\_OFF);  
digitalWrite(segE, SEGMENT\_OFF);  
digitalWrite(segF, SEGMENT\_OFF);  
digitalWrite(segG, SEGMENT\_OFF);  
break;  
}  
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -19**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/\*  
\* Show messages on an 8x8 led matrix,  
#include <FrequencyTimer2.h>  
#define SPACE { \  
{0, 0, 0, 0, 0, 0, 0, 0}, \  
{0, 0, 0, 0, 0, 0, 0, 0}, \  
{0, 0, 0, 0, 0, 0, 0, 0}, \  
{0, 0, 0, 0, 0, 0, 0, 0}, \  
{0, 0, 0, 0, 0, 0, 0, 0}, \  
{0, 0, 0, 0, 0, 0, 0, 0}, \  
{0, 0, 0, 0, 0, 0, 0, 0}, \  
{0, 0, 0, 0, 0, 0, 0, 0} \  
} #  
define H { \  
{0, 1, 0, 0, 0, 0, 1, 0}, \  
{0, 1, 0, 0, 0, 0, 1, 0}, \  
{0, 1, 0, 0, 0, 0, 1, 0}, \  
{0, 1, 1, 1, 1, 1, 1, 0}, \  
{0, 1, 0, 0, 0, 0, 1, 0}, \  
{0, 1, 0, 0, 0, 0, 1, 0}, \  
{0, 1, 0, 0, 0, 0, 1, 0}, \  
{0, 1, 0, 0, 0, 0, 1, 0} \  
} #  
define E { \  
{0, 1, 1, 1, 1, 1, 1, 0}, \  
{0, 1, 0, 0, 0, 0, 0, 0}, \  
{0, 1, 0, 0, 0, 0, 0, 0}, \  
{0, 1, 1, 1, 1, 1, 1, 0}, \  
{0, 1, 0, 0, 0, 0, 0, 0}, \  
{0, 1, 0, 0, 0, 0, 0, 0}, \  
{0, 1, 0, 0, 0, 0, 0, 0}, \  
{0, 1, 1, 1, 1, 1, 1, 0} \  
} #  
define L { \  
{0, 1, 0, 0, 0, 0, 0, 0}, \  
{0, 1, 0, 0, 0, 0, 0, 0}, \  
{0, 1, 0, 0, 0, 0, 0, 0}, \  
{0, 1, 0, 0, 0, 0, 0, 0}, \  
{0, 1, 0, 0, 0, 0, 0, 0}, \  
{0, 1, 0, 0, 0, 0, 0, 0}, \  
{0, 1, 0, 0, 0, 0, 0, 0}, \  
{0, 1, 1, 1, 1, 1, 1, 0} \  
} #  
define O { \  
{0, 0, 0, 1, 1, 0, 0, 0}, \  
{0, 0, 1, 0, 0, 1, 0, 0}, \  
{0, 1, 0, 0, 0, 0, 1, 0}, \  
{0, 1, 0, 0, 0, 0, 1, 0}, \  
{0, 1, 0, 0, 0, 0, 1, 0}, \  
{0, 1, 0, 0, 0, 0, 1, 0}, \  
{0, 0, 1, 0, 0, 1, 0, 0}, \  
{0, 0, 0, 1, 1, 0, 0, 0} \  
} b  
yte col = 0;  
byte leds[8][8];  
// pin[xx] on led matrix connected to nn   
on Arduino  
(-1 is dummy to make array start at pos 1)  
int pins[17]= {-1, 5, 4, 3, 2, 14, 15, 16, 17,   
13, 12, 11, 10, 9, 8, 7, 6};  
// col[xx] of leds = pin yy on led matrix  
int cols[8] = {pins[13], pins[3], pins[4],   
pins[10], pins[06],   
pins[11], pins[15], pins[16]};  
// row[xx] of leds = pin yy on led matrix  
int rows[8] = {pins[9], pins[14], pins[8],  
pins[12], pins[1],   
pins[7], pins[2], pins[5]};  
const int numPatterns = 6;  
byte patterns[numPatterns][8][8] = {  
H,E,L,L,O,SPACE  
};  
int pattern = 0;

void setup() {  
// sets the pins as output  
for (int i = 1; i <= 16; i++) {  
pinMode(pins[i], OUTPUT);  
}  
// set up cols and rows  
for (int i = 1; i <= 8; i++) {  
digitalWrite(cols[i - 1], LOW);  
}  
for (int i = 1; i <= 8; i++) {  
digitalWrite(rows[i - 1], LOW);  
}  
clearLeds();

// Turn off toggling of pin 11  
FrequencyTimer2::disable();  
// Set refresh rate (interrupt timeout period)  
FrequencyTimer2::setPeriod(2000);  
// Set interrupt routine to be called  
FrequencyTimer2::setOnOverflow(display);  
setPattern(pattern);  
} v  
oid loop() {  
pattern = ++pattern % numPatterns;  
slidePattern(pattern, 60);  
} v  
oid clearLeds() {  
// Clear display array  
for (int i = 0; i < 8; i++) {  
for (int j = 0; j < 8; j++) {  
leds[i][j] = 0;  
}  
}  
} v  
oid setPattern(int pattern) {  
for (int i = 0; i < 8; i++) {  
for (int j = 0; j < 8; j++) {  
leds[i][j] = patterns[pattern][i][j];  
}  
}  
} v  
oid slidePattern(int pattern, int del) {  
for (int l = 0; l < 8; l++) {  
for (int i = 0; i < 7; i++) {  
for (int j = 0; j < 8; j++) {  
leds[j][i] = leds[j][i+1];  
}  
}  
for (int j = 0; j < 8; j++) {  
leds[j][7] = patterns[pattern][j][0 + l];  
}  
delay(del);  
}  
} /  
/ Interrupt routine  
void display() {  
digitalWrite(cols[col], LOW);   
// Turn whole previous  
column off  
col++;  
if (col == 8) {  
col = 0;  
}  
for (int row = 0; row < 8; row++) {  
if (leds[col][7 - row] == 1) {  
digitalWrite(rows[row], LOW);  
// Turn on this led  
}  
else {  
digitalWrite(rows[row], HIGH);  
// Turn off this led  
}  
}  
digitalWrite(cols[col], HIGH); //   
Turn whole column   
on at once (for equal lighting times)  
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -20**

Arduino Code  
// Controlling a servo position using a potentiometer  
(variable resistor)   
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
#include <Servo.h>   
Servo myservo; // create servo object to control a servo   
int potpin = A2; // analog pin used to connect the potentiometer  
int val; // variable to read the value from the analog pin   
void setup()   
{   
myservo.attach(9); //  
attaches the servo on pin 9 to the servo object   
}   
void loop()   
{   
val = analogRead(potpin); //   
reads the value of the potentiometer   
(value between 0 and 1023)   
val = map(val, 0, 1023, 0, 179); //  
scale it to use it with the servo (value between 0 and 180)   
myservo.write(val); //   
sets the servo position according to the scaled value   
delay(15); // waits for the servo to get there   
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -21**

/\*  
\* IRremote: IRrecvDemo - demonstrates receiving IR  
codes with Irrecv  
\* An IR detector/demodulator must be connected to   
the input RECV\_PIN.  
\* Version 0.1 July, 2009  
\* Copyright 2009 Ken Shirriff  
\* http://arcfn.com  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
#include <IRremote.h>  
int RECV\_PIN = 2;  
IRrecv irrecv(RECV\_PIN);  
decode\_results results;  
void setup()  
{  
Serial.begin(9600);  
irrecv.enableIRIn(); // Start the receiver  
} v  
oid loop()  
{  
if (irrecv.decode(&results)) {  
Serial.println(results.value, HEX);  
irrecv.resume(); // Receive the next value  
}  
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -22**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Code  
/\*  
LiquidCrystal Library - Hello World  
Demonstrates the use a 16x2 LCD display. The LiquidCrystal  
library works with all LCD displays that are compatible with the   
Hitachi HD44780 driver. There are many of them out there, and you  
can usually tell them by the 16-pin interface.  
This sketch prints "Hello World!" to the LCD  
and shows the time.  
The circuit:  
\* LCD RS pin to digital pin 12  
\* LCD Enable pin to digital pin 11  
\* LCD D4 pin to digital pin 5  
\* LCD D5 pin to digital pin 4  
\* LCD D6 pin to digital pin 3  
\* LCD D7 pin to digital pin 2  
\* LCD R/W pin to ground  
\* 10K resistor:  
\* ends to +5V and ground  
\* wiper to LCD VO pin (pin 3)  
Library originally added 18 Apr 2008  
by David A. Mellis  
library modified 5 Jul 2009  
by Limor Fried (http://www.ladyada.net)  
example added 9 Jul 2009  
by Tom Igoe  
modified 22 Nov 2010  
by Tom Igoe  
This example code is in the public domain.  
http://www.arduino.cc/en/Tutorial/LiquidCrystal  
\*/  
// include the library code:  
#include <LiquidCrystal.h>  
// initialize the library with the numbers of the interface pins  
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);  
void setup() {  
// set up the LCD's number of columns and rows:   
lcd.begin(16, 2);  
// Print a message to the LCD.  
lcd.print("hello, world!");  
} v  
oid loop() {  
// set the cursor to column 0, line 1  
// (note: line 1 is the second row, since counting begins with 0):  
lcd.setCursor(0, 1);  
// print the number of seconds since reset:  
lcd.print(millis()/1000);  
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -23**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/\*-----( Import needed libraries )-----\*/  
#include <Stepper.h>  
/\*-----( Declare Constants, Pin Numbers )-----\*/  
//---( Number of steps per revolution of INTERNAL motor in 4-step mode )---  
#define STEPS\_PER\_MOTOR\_REVOLUTION 32   
//---( Steps per OUTPUT SHAFT of gear reduction )---  
#define STEPS\_PER\_OUTPUT\_REVOLUTION 32 \* 64 //2048   
/\*-----( Declare objects )-----\*/  
// create an instance of the stepper class, specifying  
// the number of steps of the motor and the pins it's  
// attached to  
//The pin connections need to be 4 pins connected  
// to Motor Driver In1, In2, In3, In4 and then the pins entered  
// here in the sequence 1-3-2-4 for proper sequencing  
Stepper small\_stepper(STEPS\_PER\_MOTOR\_REVOLUTION, 8, 10, 9, 11);  
/\*-----( Declare Variables )-----\*/  
int Steps2Take;  
void setup() /\*----( SETUP: RUNS ONCE )----\*/  
{/  
/ Nothing (Stepper Library sets pins as outputs)  
}/\*--(end setup )---\*/  
void loop() /\*----( LOOP: RUNS CONSTANTLY )----\*/  
{  
small\_stepper.setSpeed(1); // SLOWLY Show the 4 step sequence   
Steps2Take = 4; // Rotate CW  
small\_stepper.step(Steps2Take);  
delay(2000);  
Steps2Take = STEPS\_PER\_OUTPUT\_REVOLUTION / 2;   
// Rotate CW ½ turn  
small\_stepper.setSpeed(100);   
small\_stepper.step(Steps2Take);  
delay(1000);  
Steps2Take = - STEPS\_PER\_OUTPUT\_REVOLUTION / 2;   
// Rotate CCW 1/2 turn   
small\_stepper.setSpeed(700); // 700 a good max speed??  
Small\_stepper.step(Steps2Take);  
delay(2000);  
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -24**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//Arduino Sample Code   
int Relay = 3;   
void setup()   
{   
pinMode(13, OUTPUT);  
//Set Pin13 as output   
digitalWrite(13, HIGH);  
//Set Pin13 High   
pinMode(Relay, OUTPUT);  
//Set Pin3 as output   
}   
void loop()   
{   
digitalWrite(Relay, HIGH);   
//Turn off relay   
delay(2000);   
digitalWrite(Relay, LOW);   
//Turn on relay   
delay(2000);   
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -25**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*#define DHT11\_PIN 0 // ADC0  
byte read\_dht11\_dat()  
{  
byte i = 0;  
byte result=0;  
for(i=0; i< 8; i++)  
{  
while(!(PINC & \_BV(DHT11\_PIN))); // wait for 50us  
delayMicroseconds(30);  
if(PINC & \_BV(DHT11\_PIN))   
result |=(1<<(7-i));  
while((PINC & \_BV(DHT11\_PIN))); // wait '1' finish  
}  
return result;  
}v  
oid setup()  
{  
DDRC |= \_BV(DHT11\_PIN);  
PORTC |= \_BV(DHT11\_PIN);  
Serial.begin(9600);  
Serial.println("Ready");  
}v  
oid loop()  
{  
byte dht11\_dat[5];  
byte dht11\_in;  
byte i;// start condition  
// 1. pull-down i/o pin from 18ms  
PORTC &= ~\_BV(DHT11\_PIN);  
delay(18);  
PORTC |= \_BV(DHT11\_PIN);  
delayMicroseconds(40);  
DDRC &= ~\_BV(DHT11\_PIN);  
delayMicroseconds(40);  
dht11\_in = PINC & \_BV(DHT11\_PIN);  
if(dht11\_in)  
{  
Serial.println("dht11 start condition 1 not met");  
return;  
}  
delayMicroseconds(80);  
dht11\_in = PINC & \_BV(DHT11\_PIN);  
if(!dht11\_in)  
{  
Serial.println("dht11 start condition 2 not met");  
return;  
}  
delayMicroseconds(80);// now ready for data reception  
for (i=0; i<5; i++)  
dht11\_dat[i] = read\_dht11\_dat();  
DDRC |= \_BV(DHT11\_PIN);  
PORTC |= \_BV(DHT11\_PIN);  
byte dht11\_check\_sum = dht11\_dat[0]+dht11\_dat[1]+dht11\_dat[2]+dht11\_dat[3];  
// check check\_sum  
if(dht11\_dat[4]!= dht11\_check\_sum)  
{  
Serial.println("DHT11 checksum error");  
}  
Serial.print("Current humdity = ");  
Serial.print(dht11\_dat[0], DEC);  
Serial.print(".");  
Serial.print(dht11\_dat[1], DEC);  
Serial.print("% ");  
Serial.print("temperature = ");  
Serial.print(dht11\_dat[2], DEC);  
Serial.print(".");  
Serial.print(dht11\_dat[3], DEC);  
Serial.println("C ");  
delay(2000);  
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -26**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Example sketch for interfacing with the DS1302 timekeeping chip.  
\*/  
#include <stdio.h>  
#include <string.h>  
#include <DS1302.h>  
/\* Set the appropriate digital I/O pin connections \*/  
uint8\_t CE\_PIN = 5; //Pin RST  
uint8\_t IO\_PIN = 6; //Pin DAT  
uint8\_t SCLK\_PIN = 7; //Pin CLK  
/\* Create buffers \*/  
char buf[50];  
char day[50];  
/\* Create a DS1302 object \*/  
DS1302 rtc(CE\_PIN, IO\_PIN, SCLK\_PIN);  
void print\_time()  
{  
/\* Get the current time and date from the chip \*/  
Time t = rtc.time();  
/\* Name the day of the week \*/  
memset(day, 0, sizeof(day)); /\* clear day buffer \*/  
switch (t.day) {  
case 1:  
strcpy(day, "Sunday");  
break;  
case 2:  
strcpy(day, "Monday");  
break;  
case 3:  
strcpy(day, "Tuesday");  
break;  
case 4:  
strcpy(day, "Wednesday");  
break;  
case 5:  
strcpy(day, "Thursday");  
break;  
case 6:  
strcpy(day, "Friday");  
break;  
case 7:  
strcpy(day, "Saturday");  
break;  
}  
/\* Format the time and date and insert into the temporary buffer \*/  
snprintf(buf, sizeof(buf), "%s %02d/%02d/%04d %02d:%02d:%02d",  
day,  
t.date, t.mon, t.yr,  
t.hr, t.min, t.sec);  
/\* Print the formatted string to serial so we can see the time \*/  
Serial.println(buf);  
}v  
oid setup()  
{  
Serial.begin(9600);  
/\* Initialize a new chip by turning off write protection and clearing the  
clock halt flag. These methods needn't always be called.  
See the DS1302  
datasheet for details. \*/  
//rtc.write\_protect(false);  
//rtc.halt(false);  
/\* Make a new time object to set the date and time \*/  
/\* Tuesday, May 19, 2009 at 21:16:37. \*/  
//Time t(2013, 7, 29, 23, 14, 00, 2);  
/\* Set the time and date on the chip \*/  
//rtc.time(t);  
} /  
\* Loop and print the time every second \*/  
void loop()  
{  
print\_time();  
delay(1000);  
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -27**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

VCC to arduino 5v

GND to arduino GND

Echo to Arduino pin 7

Trig to Arduino pin 8

#define echoPin 7 // Echo Pin

#define trigPin 8 // Trigger Pin

#define LEDPin 13 // Onboard LED

int maximumRange = 200; // Maximum range needed

int minimumRange = 0; // Minimum range needed

long duration, distance; // Duration used to calculate distance

void setup() {

Serial.begin (9600);

pinMode(trigPin, OUTPUT);

pinMode(echoPin, INPUT);

pinMode(LEDPin, OUTPUT); // Use LED indicator (if required)

}

void loop() {

/\* The following trigPin/echoPin cycle is used to determine the

distance of the nearest object by bouncing soundwaves off of it. \*/

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH);

//Calculate the distance (in cm) based on the speed of sound.

distance = duration/58.2;

if (distance >= maximumRange || distance <= minimumRange){

/\* Send a negative number to computer and Turn LED ON

to indicate "out of range" \*/

Serial.println("-1");

digitalWrite(LEDPin, HIGH);

}

else {

/\* Send the distance to the computer using Serial protocol, and

turn LED OFF to indicate successful reading. \*/

Serial.println(distance);

digitalWrite(LEDPin, LOW);

}

//Delay 50ms before next reading.

delay(50);

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

}

**Project -28**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

///Arduino Sample Code

void setup()

{

  Serial.begin(9600); //Set serial baud rate to 9600 bps

}

void loop()

{

int val;

val=analogRead(0);//Read Gas value from analog 0

Serial.println(val,DEC);//Print the value to serial port

delay(100);

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -29**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

int JoyStick\_X = 0; //x

int JoyStick\_Y = 1; //y

int JoyStick\_Z = 3; //key

  void setup()

{

  pinMode(JoyStick\_Z, INPUT);

  Serial.begin(9600); // 9600 bps

}

void loop()

{

  int x,y,z;

  x=analogRead(JoyStick\_X);

  y=analogRead(JoyStick\_Y);

  z=digitalRead(JoyStick\_Z);

  Serial.print(x ,DEC);

  Serial.print(",");

  Serial.print(y ,DEC);

  Serial.print(",");

  Serial.println(z ,DEC);

  delay(100);

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -30**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*byte sensorPin = 3;

byte indicator = 13;

void setup()

{

pinMode(sensorPin,INPUT);

pinMode(indicator,OUTPUT);

Serial.begin(9600);

}

void loop()

{

byte state = digitalRead(sensorPin);

digitalWrite(indicator,state);

if(state == 1)Serial.println("Somebody is in this area!");

else if(state == 0)Serial.println("No one!");

delay(500);

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Project -31**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <Wire.h>

// Registers for ADXL345

#define ADXL345\_ADDRESS (0xA6 >> 1) // address for device is 8 bit but shift to the

// right by 1 bit to make it 7 bit because the

// wire library only takes in 7 bit addresses

#define ADXL345\_REGISTER\_XLSB (0x32)

int accelerometer\_data[3];

// void because this only tells the cip to send data to its output register

// writes data to the slave's buffer

void i2c\_write(int address, byte reg, byte data) {

// Send output register address

Wire.beginTransmission(address);

// Connect to device

Wire.write(reg);

// Send data

Wire.write(data); //low byte

Wire.endTransmission();

}

// void because using pointers

// microcontroller reads data from the sensor's input register

void i2c\_read(int address, byte reg, int count, byte\* data) {

// Used to read the number of data received

int i = 0;

// Send input register address

Wire.beginTransmission(address);

// Connect to device

Wire.write(reg);

Wire.endTransmission();

// Connect to device

Wire.beginTransmission(address);

// Request data from slave

// Count stands for number of bytes to request

Wire.requestFrom(address, count);

while(Wire.available()) // slave may send less than requested

{

char c = Wire.read(); // receive a byte as character

data[i] = c;

i++;

}

Wire.endTransmission();

}

void init\_adxl345() {

byte data = 0;

i2c\_write(ADXL345\_ADDRESS, 0x31, 0x0B); // 13-bit mode +\_ 16g

i2c\_write(ADXL345\_ADDRESS, 0x2D, 0x08); // Power register

i2c\_write(ADXL345\_ADDRESS, 0x1E, 0x00); // x

i2c\_write(ADXL345\_ADDRESS, 0x1F, 0x00); // Y

i2c\_write(ADXL345\_ADDRESS, 0x20, 0x05); // Z

// Check to see if it worked!

i2c\_read(ADXL345\_ADDRESS, 0X00, 1, &data);

if(data==0xE5)

Serial.println("it work Success");

else

Serial.println("it work Fail");

}

void read\_adxl345() {

byte bytes[6];

memset(bytes,0,6);

// Read 6 bytes from the ADXL345

i2c\_read(ADXL345\_ADDRESS, ADXL345\_REGISTER\_XLSB, 6, bytes);

// Unpack data

for (int i=0;i<3;++i) {

accelerometer\_data[i] = (int)bytes[2\*i] + (((int)bytes[2\*i + 1]) << 8);

}

}

// initialise and start everything

void setup() {

Wire.begin();

Serial.begin(9600);

for(int i=0; i<3; ++i) {

accelerometer\_data[i] = 0;

}

init\_adxl345();

}

void loop() {

read\_adxl345();

Serial.print("ACCEL: ");

Serial.print(float(accelerometer\_data[0])\*3.9/1000);//3.9mg/LSB scale factor in 13-bit mode

Serial.print("\t");

Serial.print(float(accelerometer\_data[1])\*3.9/1000);

Serial.print("\t");

Serial.print(float(accelerometer\_data[2])\*3.9/1000);

Serial.print("\n");

delay(100);

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*