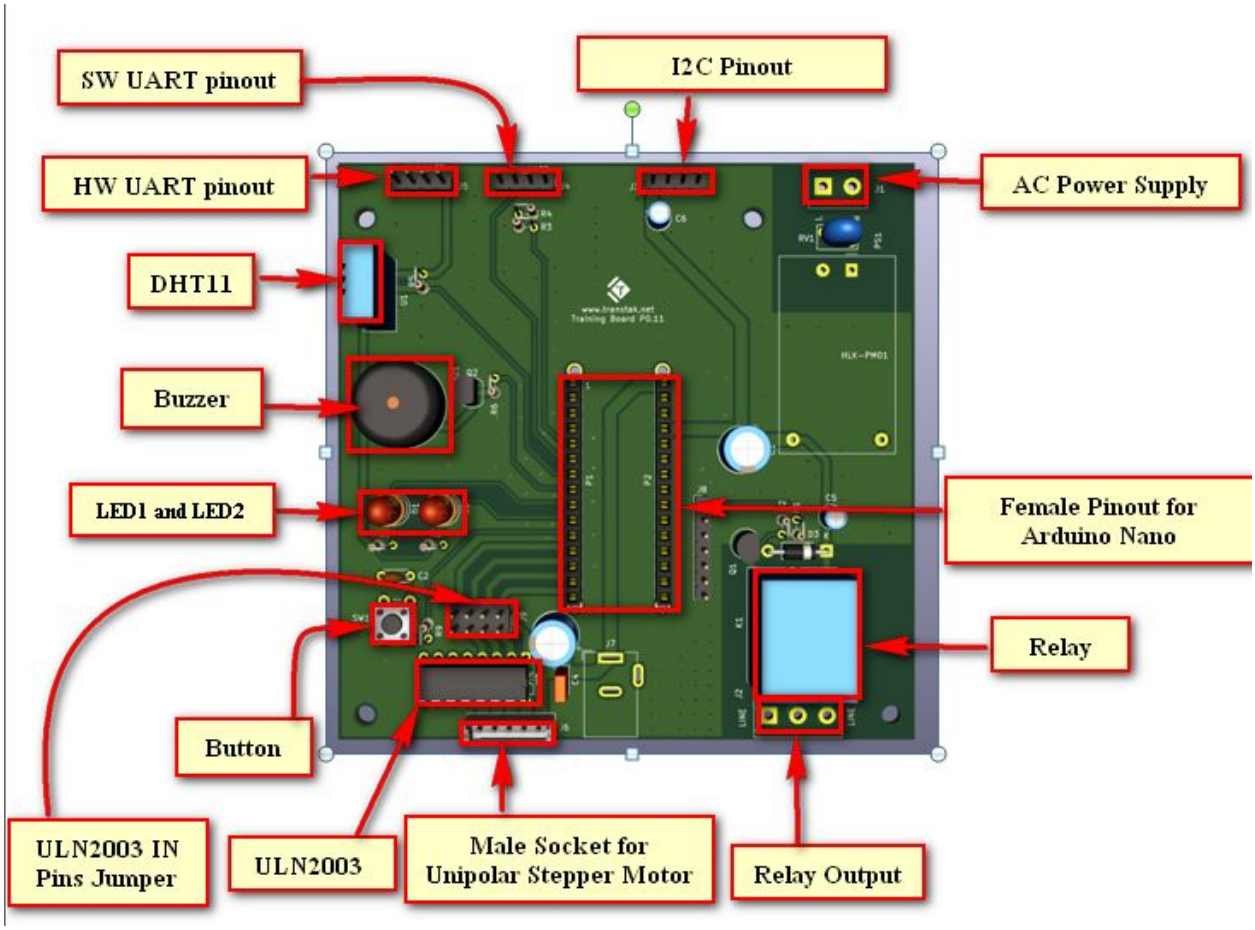
Hardware Overview



LED

ဒီ shiled မှာဆိုရင် build in LED နှစ်လုံးကို arduino nano ရဲ့ digital pin နှစ်ပင် ချိတ်ဆက်ပေးထားပါတယ်။ အောက်မှာ အသုံးပြုတဲ့ pin number နဲ့ LED နှစ်လုံးကို တစ်လှည့်စီ တစ်စက္ကန့်ခြား တခါ blink ဖြစ်နေစေမယ့် example sketch တစ်ခု ဖော်ပြထားပါတယ်။

Pin description (In PCB – Not required external hardware)

LED1	**D6**
LED2	**D7**

Example code

```
//--- Begin code (copy from here)---
void setup() {
  // put your setup code here, to run once:
  pinMode(6,OUTPUT);//initialize the led1 pin as OUTPUT
  pinMode(7,OUTPUT);//initialize the led2 pin as OUTPUT
}
void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(6,HIGH);//LED1 ON
  delay(1000);//LED1 ON 1 second
  digitalWrite(6,LOW);//LED1 OFF
  digitalWrite(7,HIGH);//LED2 ON
  delay(1000);//LED2 ON 1 second
  digitalWrite(7,LOW);//LED2 OFF
}//--- End code (copy to here)---
```

Code ရှင်းလင်းချက်

```
void setup(){
        //code
        }
void loop(){
        //code
        }
```

Arduino မှာ အခြေခံ structure အနေနဲ့ void setup နဲ့ void loop ဆိုတဲ့ function နှစ်ခု ရှိပါတယ်။ setup function ရဲ့ လုပ်ဆောင်ချက်ကတော့ သူ့ထဲမှာ ရေးထားတဲ့ code တွေကို တစ်ကြိမ်တစ်ခါပဲ run ပေးမှာ ဖြစ်ပါတယ်။ အသုံးပြုမယ့် pin နံပါတ်တွေ သတ်မှတ်ခြင်း၊ တစ်ကြိမ်တစ်ခါပဲ run ချင်တဲ့ code တွေကို setup function ထဲမှာ ရေးလေ့ရှိပါတယ်။

loop function ရဲ့ လုပ်ဆောင်ချက်ကတော့ သူ့ထဲမှာ ရေးထားသမျှ code တွေကို ထပ်ခါထပ်ခါ sequence အလိုက် အထက်မှ အောက်သို့ အချိန်တိုင်း loop ပတ်ပြီး run ပေးနေမှာ ဖြစ်ပါတယ်။

pinMode(6,OUTPUT);

pinMode(7,OUTPUT);

Arduino ရဲ့ digital I/O pin များကို အသုံးမပြုခင်မှာ အသုံးပြုမယ့် pin နံပါတ်နဲ့ Input အနေနဲ့ သုံးမလား Output အနေနဲ့ သုံးမလားဆိုတာကို သတ်မှတ်ပေးရပါမယ်။ အဲ့လို သတ်မှတ်ပေးဖို့ pinMode function ကို သုံးရပါတယ်။

pinMode function ရဲ့ syntax ကတော့

pinMode(pinNumber, Mode);

ဒီ example ဆိုရင် Led နှစ်ခုရဲ့ anode တွေကို arduino ရဲ့ digital pin 6 နဲ့ 7 ကို ချိတ်ဆက်ထားပြီး Led လင်းဖို့အတွက် 5V ထုတ်ပေးရမှာဖြစ်တဲ့အတွက် OUTPUT အနေနဲ့ အသုံးပြုရမှာပါ။

pinMode သတ်မှတ်ပြီးပါက တကယ်လုပ်ဆောင်ချက် OUTPUT ထုတ်ခိုင်းရမှာဖြစ်ပါတယ်။ digital pin တွေကို OUTPUT ထုတ်ခိုင်းချင်ပြီဆိုရင်တော့ digitalWrite() function ကို သုံးရမှာဖြစ်ပါတယ်။ digitalWrite function ရဲ့ syntax ကတော့

digitalWrite(pinNumber, value);

Parameter တွေကတော့

pin: used pin number

value: HIGH or LOW

HIGH ဆိုရင်တော့ 5V ထုတ်ပေးမှာဖြစ်ပြီး LOW ဆိုရင်တော့ 0V ဖြစ်ပါတယ်။ LED ကို ဖွင့်လိုက်ပိတ်လိုက်လူပ်ချင်တာ ဖြစ်တဲ့အတွက် digitalWrite function မှာ တစ်လှည့်ဆီ HIGH LOW ပေးပေးရမှာ ဖြစ်ပါတယ်။

LED ကို Blink ဖြစ်နေဖို့အတွက်တော့ လင်းတဲ့အချိန်နဲ့ ပိတ်တဲ့အချိန် တစ်ပြိုင်တည်းမဖြစ်နေဖို့ အချိန်တစ်ခုခြားပေးရမှာဖြစ်ပါတယ်။ အဲ့တာမှ လင်းတဲ့အချိန်တစ်ခု ပိတ်တဲ့အချိန်တစ်ခုနဲ့ လင်းလား ပိတ်လားဆိုတာ တိတိကျကျသိရမှာ ဖြစ်ပါတယ်။ အဲ့တာကြောင့် delay () function ကို digitalWrite တွေရဲ့ ကြားမှာ ညှပ်သုံးပေးရမှာဖြစ်ပါတယ်။ delay function ကို သုံးတဲ့အခါမှာ program ဟာ အချိန်တစ်ခု (milisecond) pauses ဖြစ်သွားမှာဖြစ်ပါတယ်။ delay function ရဲ့ syntax ကတော့

delay(ms);

Example မှာ မီးလင်းတစ်ကွန့် ပိတ်တစ်စကွန့် ခြားပြီး blink ဖြစ်နေချင်တာမို့ 1000 ကို သုံးထားတာ ဖြစ်ပါတယ်။ mili ရဲ့ တန်ဖိုးက 10^-3 မို့လို့ပါ။

Relay

Relay အတွက် program ရေးသားခြင်းကလည်း LED အတွက် ရေးသားခြင်းနဲ့ တူပါတယ်။ Relay အတွက် OUTPUT AC ကြိုးနဲ့ ချိတ်ဖို့ အတွက် Screw Terminal (J2) မှာ Normally Open (NO), Common, Normally Close (NC) ဆိုပြီး Terminal သူးခု ပါဝင်ပါတယ်။

Relay ရဲ့ ကွိုင်ကပ်ဖို့ pulse ပေးမယ့် 5V DC ကိုတော့ Arduino ရဲ့ digital pin တစ်ခုနဲ့ ချိတ်ဆက်ထားပါတယ်။

Relay ကို 5V pulse ပေးလျှင် ကွိုင်ကပ်သွားပြီး NO က close ဖြစ်သွားပြီး NC က open ဖြစ်သွားမှာ ဖြစ်ပါတယ်။ pulse မရှိတော့လျှင် နဂိုမူလအခြေအနေကို ပြန်လည်ရောက်ရှိသွားမှာ ဖြစ်ပါတယ်။ Relay ကို အများအားဖြင့် AC ပစ္စည်းများမှာ အသုံးပြုသလို DC ပစ္စည်းများဖြင့်လည်း switch သဘောမျိုး ချိတ်ဆက်အသုံးပြုနိုင်ပါတယ်။

အောက်မှာ Relay ကို အသုံးပြုပြီး AC မီးသီး တစ်လုံး ဖွင့်ပိတ် ပြုလုပ်ပုံကို pinout,example code များနှင့် တကွ ရှင်းပြထားပါတယ်။

Pin Description

Input pin (in PCB)	**A3**
Output pin(for external hardware)	**NO, Common, NC**

Example code

```
//--- Begin code (copy from here)---

void setup() {

  // put your setup code here, to run once:

  pinMode(A3,OUTPUT);//initialize the relay pin as OUTPUT

}

void loop() {
```

```
  // put your main code here, to run repeatedly:
  digitalWrite(A3,HIGH);//NO->close, NC->open
  delay(1000);//delay 1 second
  digitalWrite(A3,LOW);//NO->open, NC->close(normal state)
  delay(1000);//delay 1 second
}
//--- End code (copy to here)---
```

Code ရှင်းလင်းချက်

LED blink example မှာ ရှင်းထားခဲ့တဲ့ function တွေကိုပဲ အသုံးပြုထားတာဖြစ်လို့ သဘောတရား အတူတူပဲ ဖြစ်ပါတယ်။ အသုံးပြုမယ့် pin number လေးတွေပဲ ပြောင်းလဲ သွားတာ ဖြစ်ပါတယ်။

Push Button

Push Button ကိုတော့ arduino မှာဆိုရင် digital Input အနေနဲ့ သုံးပါတယ်။ ခလုတ်ကို ဖိထားတဲ့အခါ circuit ကို ဖျက်ထားတဲ့ connection contact ဖြစ်သွားပြီး curcuit တစ်ပတ်ပြည့်ပါတယ်။ Button ကို အသုံးပြုပြီး board မှာ ပါဝင်တဲ့ LED1 မီးလုံးကို ခလုတ်ကိုဖိရင်လင်းနေစေပြီး လွှတ်လိုက်တဲ့အခါ မီးပိတ်သွားစေမယ့် Example code လေးကို ဖော်ပြထားပါတယ်။

Pin Description (In PCB – Not required external hardware)

Button Signal Pin	D8

Example Sketch

```
//--- Begin code (copy from here)---
```

```
int buttonPin = 8; //pin number connect with button pin
int ledPin = 6; //pin number connect with LED1 pin
int buttonState = 0; //variable for reading the button status
void setup() {
  // put your setup code here, to run once:
  // initialize the LED pin as an output:
  pinMode(ledPin,OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
}
void 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);
  }
}
```

//--- End code (copy to here)---

Code ရှင်းလင်းချက်

Arduino ရဲ့ digital pin တွေကို OUTPUT အနေနဲ့ အသုံးပြုဖို့ကို အပေါ်မှာ ရှင်းလင်းခဲ့ပြီး ဖြစ်ပါတယ်။ ဒီတစ်ခါတော့ INPUT အနေနဲ့ အသုံးပြုမှာ ဖြစ်ပါတယ်။ ပြင်ပကနေ pin ခံနိုင်ရည်ရှိတဲ့ voltage ဖြစ်တဲ့ 5V ဝင်လာလား မဝင်လာလား ဆိုတာကို စောင့်ကြည့်မှာဖြစ်ပါတယ်။ Pushbutton ဟာ arduino ရဲ့ 5V pin နဲ့ ချိတ်ဆက်ထားပြီး တစ်ဖက်မှာ arduino ရဲ့ digital pin နဲ့ ချိတ်ဆက်ထားမှာ ဖြစ်ပါတယ်။ push button မနှိပ်ခင်မှာ 5V နဲ့ connection မဖြစ်သေးပဲ pull down resistor ကြောင့် 0V အခြေအနေ or LOW အခြေအနေဖြစ်နေမှာပါ။ push button နှိပ်တဲ့အခါ 5V နဲ့ connection ဖြစ်သွားပြီး arduino ရဲ့ digital pin ထဲကို Input pulse အနေနဲ့ ဝင်သွားမှာ ဖြစ်ပါတယ်။ Arduino pin ကို Input အနေနဲ့ သုံးမှာဖြစ်လို့ pinMode function မှာ button နဲ့ ချိတ်ဆက်ထားတဲ့ pin ကို INPUT လို့ပေးထားတာ ဖြစ်ပါတယ်။ HIGH ဝင်လာလား LOW ဝင်လာလားဆိုတဲ့ အခြေအနေကို သိဖို့အတွက် digitalRead() function ကိုသုံးရမှာဖြစ်ပါတယ်။ digitalRead ရဲ့ syntax ကတော့

digitalRead(pinNumber);

digitalRead function ဟာ သူ့ထဲမှာ ပေးထားတဲ့ pin number ရဲ့ HIGH or LOW အခြေအနေ တနည်းအားဖြင့် 1 or 0 အခြေအနေကို return ပြန်ထုတ်ပေးမှာဖြစ်ပါတယ်

Variable

C program တွေမှာ variable တွေကို interger, float, character တန်ဖိုးတွေကို သိမ်းထားချင်တဲ့အခါ သုံးပါတယ်။

Interger = ကိန်းပြည့်ဂဏန်း

Float = ဒသမကိန်း

character = ABCD.... စသဖြင့် အက္ခရာများ

ဒီတန်ဖိုးတွေကို program ထဲမှာ သုံးချင်တယ်ဆိုရင် Variable အနေနဲ့ တည်ပေး (declare) လုပ်ပေးရပါတယ်။ Variable ထားခြင်းအားဖြင့် program ထဲမှာ ပြင်စရာလိုလာတဲ့အခါမှာ program တစ်ခုလုံးလိုက်ပြင်စရာ မလိုပဲ declare လုပ်ခဲ့တဲ့နေရာမှာပဲ သွားပြင်လို့ရမှာဖြစ်ပါတယ်။ မြင်သာအောင် ပြောရင် ကိန်းဂဏန်း သိုမဟုတ် အက္ခရာတန်ဖိုးလေးတွေကို ခွက်တစ်ခုထဲမှာ ထည့်ပြီး အဲ့ခွက်ကို နာမည်ပေးခြင်းနဲ့တူပါတယ်။ Variable တွေကို အောက်ပါ syntax နဲ့ declare လုပ်ပါတယ်။

dataType variableName = value;

int **var** = **10**;

float num = **1.23**;

char abc = '**a**';

Declare လုပ်တဲ့အခါ အရင်ဆုံး သိမ်းဆည်းမယ် data ရဲ့ အမျိုးအစား data type ကို ကြေညာရပါတယ်။ ပီးရင် variable name ပေးရပါတယ်။ variable name ကို မိမိကြိုက်နှစ်သက်ရာပေးလို့ရပါတယ်။ အဲ့ဒီနောက် = (assigment operator) ကို သုံးပြီး သိမ်းဆည်းမယ့် value ကို ထည့်ပြီး assign လုပ်ပါတယ်။ variable ကို အသုံးပြုခြင်းအားဖြင့် variable name လိုကိုပဲ ခေါ်ပြီး ကြိုက်တဲ့ value ကို ခေါ်သုံးနိုင်မှာဖြစ်ပါတယ်။

Push Button example program မှာတော့

```
int buttonPin = 8;

int ledPin = 6;

int buttonState = 0;
```

Buutton နဲ့ ချိတိဆက်ထားတဲ့ arduino ရဲ့ pin number နှင့် led နဲ့ ချိတ်တက်ထားတဲ့ arduino ရဲ့ pin number များကိုို variable အဖြစ် declare လုပ်ပါတယ်။ button pin ကို စောင့်ကြည်ပီး LOW ဖြစ်လား HIGH ဖြစ်လား သိုမဟုတ် 1 or 0 ဆိုတာကို စောင့်ကြည့်မယ့် button နှိပ်မနှိပ်ကို သိမ်းဆည်းပေးမယ့် buttonState variable တန်ဖိုးကိုလည်း 0 အနေနဲ့ initilize လုပ်ထားခြင်းဖြစ်ပါတယ်။ button မနှိပ်ပဲ 1 မဖြစ်အောင် initilize လုပ်ရခြင်းဖြစ်ပါတယ်။

setup function ထဲမှာတော့ ledpin ကို OUTPUT, button pin ကို INPUT အနေနဲ့ အသီးသီးကြေငြာထားပါတယ်။

ခလုတ်နှိပ်လိုက်ရင် မီးလင်းနေပြီး မနှိပ်တဲ့အခါ မီးပိတ်နေဖို့ ရေးမှာဖြစ်တဲ့အတွက် နှိပ်တယ် မနှိပ်ဘူးဆိုတဲ့ အခြေအနေ (condition) ကို arduino က သိအောင် စစ်ပေးနေရမှာဖြစ်ပါတယ်။ C program ရေးတဲ့အခါ condition စစ်ဖို့ if statement ကို သုံးပါတယ်။ if statement ရဲ့ syntax ကတော့

```
if(condition){

  //code

}
```

if ရဲ့ ကွင်းစကွင်းပိတ်ထဲမှာ စစ်မယ့် condition ကို ရေးပြီး အဲ့ condition true ဖြစ်ခဲ့ရင် if ရဲ့ {} ထဲမှာရေးထားတဲ့ code တွေ အလုပ်လုပ်မှာဖြစ်ပါတယ်။ condition တစ်ခုထဲကိုပဲ စစ်တာ မဟုတ်ပဲ ဒီ conditon တွေမှာဆို ဒီဟာ လုပ်လို့ ခိုင်းချင်ရင်တော့ if-else ကို သုံးရမာ ဖြစ်ပါတယ်။

if - else ရဲ့ syntax ကတော့

```
if(condition){

  //code

}else{

  //code

}
```

if ထဲက condition true မဖြစ်ခဲ့ရင် else ထဲက code တွေ အလုပ်လုပ်မှာ ဖြစ်ပါတယ်။

Button example မှာတော့

buttonState = digitalRead(buttonPin);

ဆိုပြီးတော့ button နဲ့ ချိတ်ဆက်ထားတဲ့ arduino pin ရဲ့ input တန်ဖိုးကို read လုပ်ပေးနေမှာဖြစ်ပါတယ်။ digitalRead function က 1 or 0 ပြန်ထုတ်ပေးမှာဖြစ်ပါတယ်။ ထွက်လာတဲ့ 1 or 0 ကို buttonState ဆိုတဲ့ variable ထဲမှာ သွားထည့်လိုက်တာ ဖြစ်ပါတယ်။ အဲ့ buttonState တနိဖိုးကို if ထဲမှာ HIGH or LOW သွားစစ်တာဖြစ်ပါတယ်။ == ဆိုတာကတော့ ဟိုဘက်နဲ့ ဒီဘက် operand နှစ်ခု ညီလာဆိုတာ စစ်တာဖြစ်ပါတယ်။

Arduino IDE မှာ HIGH ဆိုတာနဲ့ 1 ဆိုတာတူသလို LOW ဆိုတာနဲ့ 0 ဆိုတာအတူတူပဲ ဖြစ်ပါတယ်။ integer အနေနဲ့ပဲ သိမ်းတာဖြစ်ပါတယ်။ button မနှိပ်ခင်မှာ 0 ကိုပဲ digitalRead တန်ဖိုးက return ပြန်နေပေးမှာဖြစ်လို့ buttonState ဟာ 0 ဖြစ်နေမှာပါ။ if ထဲမှာ (buttonState == HIGH) လို့ စစ်ထားတာဖြစ်လို့ (0==1) ဆိုတော့ true မဖြစ်တာကြောင့် else ထဲက Led ကို off လုပ်ထားတဲ့ code ကိုပဲ run မှာ ဖြစ်ပါတယ်။ ခလုတ်နှိပ်လိုက်လို့ digitalRead() က 1 return ပြန်သွားပြီး buttonState ဟာ 1 ဖြစ်သွားမှာ ဖြစ်ပါတယ်။ codition ထဲမှာ (1==1) ညီသွားပြီဖြစ်လို့ if statement true ဖြစ်ပြီး သူ့ထဲက Led on ပေးတဲ့ code ကိုူ run ပေးမှာ ဖြစ်ပါတယ်။

Buzzer

အသံ output အနေနဲ့ Buzzer များကို အသုံးပြုနိုင်ပါတယ်။ ဒီ board မှာတော့ buzzer အတွက် pulse and frequency ပေးမယ့် digital pin ကို arduino ရဲ့ digital pin တစ်ခုနဲ့ ချိတ်ဆက်ထားပါတယ်။ အောက်မှာ buzzer ကို အရေးပေါ် ambulance အသံပေါ်အောင် tone function ကို သုံးပြီး ရေးထားတဲ့ Example Sketch ကို ပြသပေးထားပါတယ်။

Pin description (In PCB – Not required external hardware)

Buzzer pin	**D5**

Example Sketch

```
//--- Begin code (copy from here)---

int buzzer = 5;//variable for buzzer pin

void setup() {

  // put your setup code here, to run once;

 }
```

```
void loop() {
  // put your main code here, to run repeatedly:
  tone(buzzer,400); //tone(pinNumber,frequency)
  delay(500);
  noTone(buzzer);//noTone(pinNumber)
  delay(500);
}
//--- End code (copy to here)---
```

Code ရှင်းလင်းချက်

Buzzer ကို အသံမြည်အောင် buzzer pin ကို frequency တွေ ပေးရမှာဖြစ်လို့ tone() နဲ့ notone() ဆိုတဲ့ function ကို သုံးထားပါတယ်။ tone() function ဟာ လိုအပ်တဲ့ frequency ကို လိုအပ်တဲ့ pin ပေါ်မှာ ထုတ်ပေးအောင် သုံးလို့ရပါတယ်။ အချိန် (duration) ကိုလည်း လိုအပ်သလို သတ်မှတ်ပေးလို့ရသလို notone() function မခေါ်ခင်ထိ frequency ကို ပေးလွှတ်နေမှာဖြစ်ပါတယ်။ tone() function ဟာ တစ်ကြိမ်မှာ tone တစ်ခုကိုပဲ ထုတ်ပေးနိုင်ပါတယ်။ အခြား pin တစ်ခုမှာ tone function သုံးထားရင် နောက်တစ်ခုမှာ သုံးလို့ရမှာ မဟုတ်ပါဘူး။

tone() function ရဲ့ syntax ကတော့

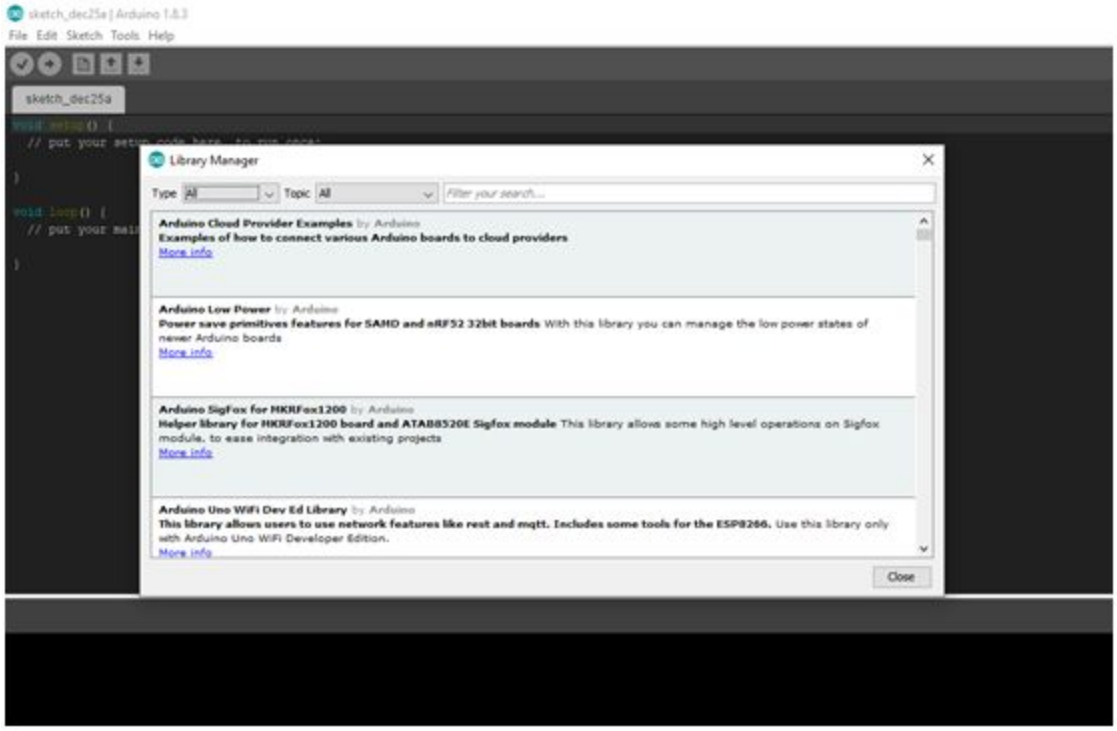
```
tone(pin,frequency);
tone(pin,frequeny,duration);
```

pin ကတော့ buzzer နဲ့ချိတ်ဆက်ထားပြီး frequency ထုတ်ပေးရမယ့် arduino ရဲ့ pin number ပါ။ frequency ကတော့ ထုတ်ပေးရမယ့် ကြိမ်နှုန်း hertz တန်ဖိုး ဖြစ်ပါတယ်။ unsinged int datatype ပါ။ duration ကတော့ tone ထုတ်ပေးရမယ့် အချိန်ဖြစ်ပါတယ်။

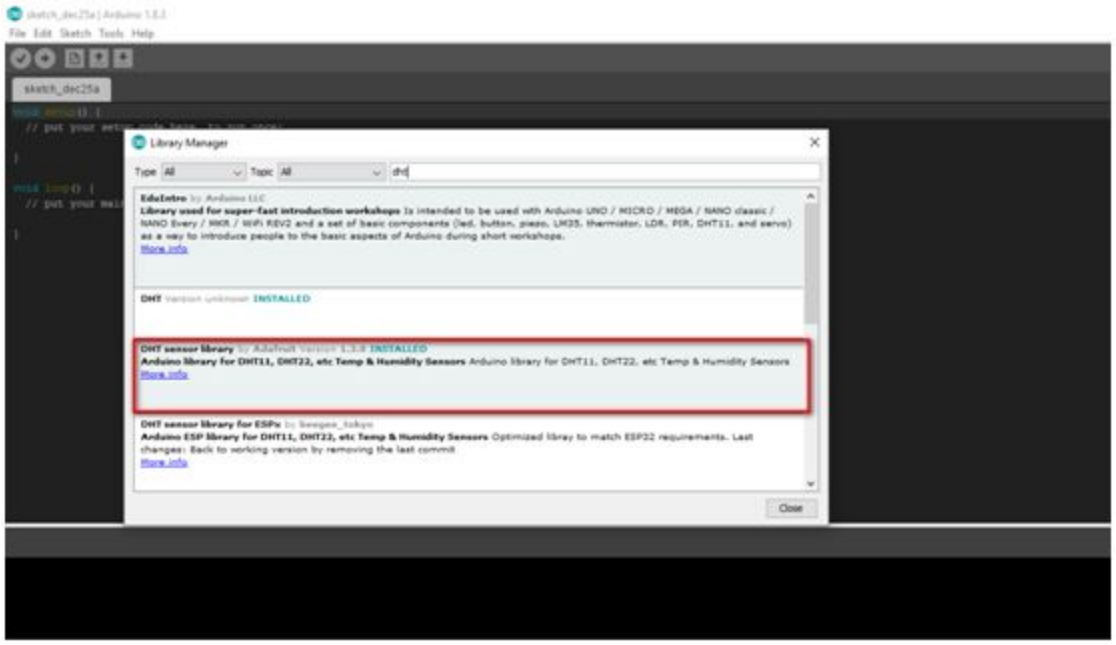
DHT11 Temperature and Humidity Sensor

DHT11 Sensor ကို အသုံးပြုဖို့အတွက် Arduino Library နှစ်ခု download ပြုလုပ်ဖို့ လိုအပ်ပါတယ်။
လိုအပ်တဲ့ Library တွေက
1. DHT library

2. Adafruit Unified Sensor Driver library
DHT Library install ပြုလုပ်ဖို့ အတွက် Arduino IDE ကိုဖွင့်ပြီး
Step 1. Sketch>>Include Library>> Manage Libraries သို့သွားပါ။ Library Manager ပေါ်လာပါလိမ့်မယ်။



Step 2. Search box တွင် DHT ဟုရိုက်ရှာပြီး ပုံတွင်ပြထားသော Library ကို တစ်ချက် click ၍ install ကို click ပါ။



Adafruit Unified Sensor Sensor Driver library ကို download ပြုလုပ်ရန်
Step 1. အောက်တွင်ပေးထားသော Git repo link ကို နှိပ်ပါ

Link : https://github.com/adafruit/Adafruit_Sensor

Step 2. Git repo ရှိ clone and download ကို နှိပ်ပါ။
Step 3. Adafruit_Sensor_master.zip နာမည်ဖြင့် zip file တစ်ခု ရရှိလာပါမယ်။

Step 4. Arduino IDE ကို ဖွင့်ပြီး Sketch>>Include Library>>Add. ZIP Library ကို click ပြီး ခုန ရရှိလာသော zip file ကို ရွေးပေးပါ။

Step 5. Library ထည့်ပြီးသွားလျှင် arduino IDE ကို ပိတ်၍ ပြန်ဖွင့်ပါ။

DHT sensor မှာ DHT11, DHT22 , DHT21 ဆိုပြီး သုံးမျိုးရှိပါတယ်။ ယခု Board မှာတော့ DHT11 ကို အသုံးပြုထားပါတယ်။ DHT11 ရဲ့ Data ယူမယ့် pin ကို တော့ arduino nano ရဲ့ digital pin တစ်ခုနဲ့ ချိတ်ဆက်ထားတာ ဖြစ်ပါတယ်။ အောက်မှာတော့ pin description နဲ့ temperature, humidity, heat index တန်ဖိုးများကို serial monitor မှာ ပြသအောင် ရေးထားတဲ့ Example sketch တစ်ခုကို ပြသပေးထားပါတယ်။ Serial monitor ကို သုံးမှာ ဖြစ်တဲ့အတွက် run နေစဉ်မှာ computer နဲ့ arduino ကို ချိတ်ဆက်ပေးထားဖို့ လိုပါတယ်။

Pin description (In PCB – Not required external hardware)

DHT11 data pin	**D4**

Example sketch

```
//--- Begin code (copy from here)---

#include "DHT.h"

#define DHTPIN 4        // what digital pin we're connected to


// Uncomment whatever type you're using!

#define DHTTYPE DHT11   // DHT 11

//#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321

//#define DHTTYPE DHT21   // DHT 21 (AM2301)


// Initialize DHT sensor.
```

```
DHT dht(DHTPIN, DHTTYPE);
void setup() {
  // put your setup code here, to run once;
  Serial.begin(9600);
  Serial.println("DHTxx test!");
  dht.begin();
}
void loop() {
  // put your main code here, to run repeatedly:
  // Wait a few seconds between measurements.
  delay(2000);

  // Reading temperature or humidity takes about 250 milliseconds!
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  // Read temperature as Celsius (the default)
  float t = dht.readTemperature();
  // Read temperature as Fahrenheit (isFahrenheit = true)
  float f = dht.readTemperature(true);

  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) || isnan(f)) {
    Serial.println("Failed to read from DHT sensor!");
            return;
```

```
  }

  // Compute heat index in Fahrenheit (the default)
  float hif = dht.computeHeatIndex(f, h);
  // Compute heat index in Celsius (isFahreheit = false)
  float hic = dht.computeHeatIndex(t, h, false);

  Serial.print("Humidity: ");
  Serial.print(h);
  Serial.print(" %\t");
  Serial.print("Temperature: ");
  Serial.print(t);
  Serial.print(" *C ");
  Serial.print(f);
  Serial.print(" *F\t");
  Serial.print("Heat index: ");
  Serial.print(hic);
  Serial.print(" *C ");
  Serial.print(hif);
  Serial.println(" *F");
}
//--- End code (copy to here)---
```

Code ရှင်းလင်းချက်

```
#include "DHT.h"
```

DHT library ကို သုံးမယ်လို့ ကြေငြာပါတယ်။

```
#define DHTPIN 4

#define DHTTYPE DHT11
```

DHT ရဲ့ signal pin နှင့် ချိတ်ဆက်ထားတဲ့ arduino pin number နဲ့ အသုံးပြုမယ့် DHT အမျိုးအစားတို့ကို သတ်မှတ်ပေးခြင်းဖြစ်ပါတယ်။

```
DHT dht(DHTPIN, DHTTYPE);
```

dht Object ကို creat လုပ်ပီး parameter ထဲမှာ pin number နဲ့ DHT type ကို ထည့်ခြင်းဖြစ်ပါတယ်။ setup section ထဲမှာတော့

```
Serial.begin(9600);

Setrial.println("DHTxx test!");

dht.begin();
```

Debugging နဲ့ DHT ကရသော တန်ဖိုးတွေကို serial monitor ကနေ ဖတ်ရှုဖို့အတွက် Serial begin ကို buard rate 9600 နဲ့ စထားပါတယ်။ Serial monitor မှာ DHTxx test ဆိုတဲ့ စာသားလေးကို print လုပ်ပါတယ်။ DHT မှာ signal တွေ စတင်ရရှိဖို့အတွက် dht.begin() method ကို စခေါ်ပါတယ်။

Loop section ထဲမှာတော့ DHT မှ signal ကောင်းကောင်းမွန်မွန်ရရှိဖို့အတွက် 2 second လောက် delay ထားပြီးတဲ့ နောက်မှာ

```
float h=dht.readHumidity();

float t=dht.readTemperature();

float f = dht.readTemperature(true);
```

DHT မှ ရရှိလာတဲ့ Humidity တန်ဖိုးကို dht.readHumidity() method ကို သုံးပြီး ဖတ်ပါတယ်။ ရရှိလာတဲ့ တန်ဖိုးကို "h" ဆိုတဲ့ variable ထဲကို assign လုပ်ပါတယ်။

DHT မှ ရရှိလာတဲ့ Temperature တန်ဖိုးကို dht.readTemperature() method ကို သုံးပြီး ဖတ်ပါတယ်။ ရရှိလာတဲ့ တန်ဖိုးကို "t" ဆိုတဲ့ variable ထဲမှာ assign လုပ်ပါတယ်။

dht.readTemperature() method ဟာ default အနေနဲ့ Celsius(°C) တန်ဖိုးကို ထုတ်ပေးပါတယ်။ Fahrenheit တန်ဖိုးကို ရဖို့ method ရဲ့ parameter ထဲမှာ true လေး ထည့်ပေးရင် Fahrenheit တန်ဖိုး ထုတ်ပေးမှာဖြစ်ပါတယ်။ ရရှိလာတဲ့ Fahrenheit တန်ဖိုးကို "f" ဆိုတဲ့ variable ထဲကို assign လုပ်ပါတယ်။

```
if(isnan(h) || isnan(t) || isnan(f)){
```

```
Serial.println("Failed to read from DHT sensor!");

 return;

  }
```

pin လွတ်လို့ပဲဖြစ်ဖြစ် sensor မကောင်းလို့ပဲ ဖြစ်ဖြစ် read လူပ်လို့ nan တန်ဖိုးတွေပဲ ရတယ်ဆိုရင် Serial monitor မှာ sensor မှ read fail ဖြစ်ကြောင်း ပြမှာ ဖြစ်ပါတယ်။

Heat index တန်ဖိုးကို တွက်ဖို့အတွက်တော့ dht.computeHeatIndex() method ကို သုံးပါတယ်။

```
float hif = dht.computeHeatIndex(f,h);

float hic = dht.computeHeatIndex(t,h,false);
```

Fahrenheit ပေါ်မူတည်ပီး တွက်ဖို့အတွက် အပေါ်မှာ ရရှိခဲ့တဲ့ အပူချိန် (Fahrenheit) တန်ဖိုးဖြစ်တဲ့ "f" ရယ် humidity တန်းဖိုးဖြစ်တဲ့ "h" တန်ဖိုးရယ်ကို parameter အဖြစ်ထည့်ပေးရပါတယ်။ ရရှိလာတဲ့ တန်ဖိုးကို "hif" ဆိုတဲ့ variable ထဲကို assign လုပ်ပါတယ်။ Celsius ပေါ်မူတည်ပြီး တွက်ချင်ရင်တော့ အပေါ်မှာ ရရှိခဲ့တဲ့ အပူချိန် (Celsius) တန်ဖိုးဖြစ်တဲ့ "t" ရယ် humidity တန်ဖိုးဖြစ်တဲ့ "h" တန်ဖိုးရယ်အပြင် "false" ဆိုတဲ့ parameter သုံးခု ထည့်ပေးရမှာဖြစ်ပါတယ်။ ရရှိလာတဲ့ တန်ဖိုးကို "hic" ဆိုတဲ့ variable ထဲကို assign လုပ်ပါတယ်။

```
 Serial.print("Humidity: ");

  Serial.print(h);

  Serial.print(" %\t");

  Serial.print("Temperature: ");

  Serial.print(t);

  Serial.print(" *C ");

  Serial.print(f);

  Serial.print(" *F\t");

  Serial.print("Heat index: ");

  Serial.print(hic);

  Serial.print(" *C ");

  Serial.print(hif);

  Serial.println(" *F");
```

ထို့နောက်ရရှိလာတဲ့ တန်ဖိုးများကို serial monitor မှာ ယူနစ်နှင့်တကွ ပြသဖို့ serial.print() များ အသုံးပြုထားတာ ဖြစ်ပါတယ်။

Stepper Motor Control

Stepper Motor တွေဟာ Step အလိုက် လည်တာဖြစ်တဲ့အတွက် တိကျတဲ့ နေရာမှာ ရပ်နိုင်သလို တိကျတဲ့ အပတ်ရေအလိုက် ထိန်းချုပ်နိုင်မှာ ဖြစ်ပါတယ်။ ယခု Arduino nano shield မှာတော့ ULN2003A unipolar stepper motor driver ကို build in ထည့်ပေးလို့ 5V Unipolar Stepper Motor လေးကို ထိန်းချုပ် အသုံးပြုနိုင်မှာ ဖြစ်ပါတယ်။ ULN2003A ရဲ့ Input pin များကိုတော့ J9 မှာ pin တွေနဲ့ ထွက်ပေးထားသလို Arduino ရဲ့ digital pin 4 pin မှာ built in ချိတ်ဆက်ပေးထားပါတယ်။ 5V unipolar stepper motor ရဲ့ female socket နဲ့ ချိတ်ဆက်ဖို့ board J6 မှာ Male socket တစ်ခု ထည့်သွင်းပေးထားပါတယ်။ အောက်တွင် Pin description နှင့် stepper motor ကို ထိန်းချုပ်မောင်းနှင်မယ့် example code တွေ ပါဝင်ပါတယ်။

Pin description (In PCB – Not required external hardware)

Arduino Nano Digital Pin	ULN2003A
D12	IN1
D11	IN2
D10	IN3
D9	IN4

Pin Connection between Nano shield and 5V unipolar stepper motor

J6 Male Socket	5V Stepper Motor Female Socket

Example Sketch

```
//--- Begin code (copy from here)---

#include<Stepper.h>
const float STEPS_PER_REV=32;
const float GEAR_RED=64;
```

```
const float STEPS_PER_OUT_REV=STEPS_PER_REV*GEAR_RED;
int StepsRequired;
Stepper steppermotor(STEPS_PER_REV,9,11,10,12);
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:
  steppermotor.setSpeed(1);
  StepsRequired=4;
  steppermotor.step(StepsRequired);
  delay(2000);

  StepsRequired=STEPS_PER_OUT_REV/2;
  steppermotor.setSpeed(100);
  steppermotor.step(StepsRequired);
  delay(1000);

  StepsRequired=-STEPS_PER_OUT_REV/2;
  steppermotor.setSpeed(700);
  steppermotor.step(StepsRequired);
  delay(2000);
}

//--- End code (copy to here)---
```

Code ရှင်းလင်းချက်

```
#include<stepper.h>
```

Stepper library ကို အသုံးပြုဖို့ ကြေငြာပါတယ်။

```
const float STEPS_PER_REV =32;

const float GEAR_RED=64;

const float STEPS_PER_OUT_REV = STEPS_PER_REV*GEAR_RED;
```

အသုံးပြုမယ့် မော်တာရဲ့ Step per revolution က 32 ပါ။ gear အရေအတွက်ကတော့ 64 ဖြစ်ပါတယ်။ ဒါပေမယ့် သူရဲ့ ဝင်ရိုးကို တိုက်ရိုက်ချိတ်ဆက်ထားတာ မဟုတ်ပဲ Gear နဲ့ ချိတ်ဆက်ထားတာမို့လို့ ဝင်ရိုးရဲ့ အမှန်တကယ် လည်မဲ့ step အရေအတွက်ကို လိုချင်ရင် မူရင်း step per revolution တန်ဖိုးနဲ့ ချိတ်ဆက်ထားတဲ့ Gear အစိတ်တန်ဖိုးကို

မြောက်ပေးရမှာဖြစ်ပါတယ်။ မြောက်လို့ ရရှိလာတဲ့ တန်ဖိုးက မော်တာဂီယာ တစ်ပတ်တိတိလည်ဖို့ အတွက် လိုအပ်တဲ့ step အရေအတွက်ပဲ ဖြစ်ပါတယ်။ ရရှိလာတဲ့ တန်ဖိုးကို "STEPS_PER_OUT_REV" ဆိုတဲ့ variable ထဲကို assign လုပ်ပါတယ်။

```
int stepRequired;
```

မော်တာလည်တဲ့အခါမှာ ဘယ်လောက် step လည်ရမလဲဆိုတဲ့ တန်ဖိုးကို variable အနေနဲ့ အောက်မှာ သုံးမှာဖြစ်လို့ "stepRequired" ဆိုတဲ့ variable ကို declare လုပ်ထားပါတယ်။

```
Stepper steppermotor(STEPS_PER_REV,9,11,10,12);
```

Stepper class ရဲ့ steppermotor ဆိုတဲ့ object ကို creat လူပ်ပါတယ်။ ဒီ function ကို setup section ရော loop section ရောရဲ့ အပြင်ဘက် အပေါ်ဆုံးမှာ ရေးရမှာဖြစ်ပါတယ်။ သူ့ထဲမှာ ပါဝင်တဲ့ parameter အရေအတွက်က motor မှာ ပါဝင်တဲ့ wire အရေအတွက်ပေါ်မူတည်ပြီး arduino မှာ ဘယ်နှစ်ပင်သုံးလဲဆိုတာပေါ်မှာ မူတည်ပါတယ်။

သူ့ရဲ့ syntax ကတော့

```
Stepper objectName(steps,pin1,pin2);
```

```
Stepper objectName(steps,pin1,pin2,pin3,pin4);
```

"steps" ဆိုတဲ့ parameter ကတော့ မော်တာအပြည့်တပတ်လည်ဖို့လိုအပ်တဲ့ မော်တာရဲ့ step အရေအတွက် တန်ဖိုးဖြစ်ပါတယ်။ မော်တာက step အရေအတွက်တိတိကျကျ မပေးထားပဲ step တစ်ခု လည်တဲ့အခါမှာ ဘယ်လောက် degree လည်တယ်လို့ပဲ ပေးတယ်ဆိုရင် degree တန်ဖိုးကို 360 နဲ့ စားဖြင့် မော်တာ တစ်ပတ်အပြည့်လည်တဲ့အခါမှာ ရှိမယ့် step အရေအတွက်ကို ရရှိနိုင်မှာဖြစ်ပါတယ်။ ဥပမာ 3.6° လို့ပေးထားရင် 360/3.6 = 100 step ဖြစ်ပါတယ်။ မော်တာမှာ ပါဝင်တဲ့ wire အရေအတွက်ပေါ်မူတည်ပြီး arduino ပေါ်မှာ ချိတ်ဆက်ရမယ့် pin number တွေကို သတ်မှတ်ပေးရပါတယ်။ setup section ထဲမှာတော့ ဘာမှရေးစရာမလိုပါဘူး။

```
steppermotor.setSpeed(1);
```

Motor လည်မယ့် speed (rotation per minute(RPMS)) ကို သတ်မှတ်ပေးတာဖြစ်ပါတယ်။ ဒီ setspeed() method ဟာ မော်တာလည်အောင်လုပ်ပေးတာမဟုတ်ပါဘူး။ speed ကိုပဲ သတ်မှတ်ပေးတာဖြစ်ပါတယ်။ သူရဲ့ syntax ကတော့

```
objectName.setSpeed(rpms);
```

Parameter အနေနဲ့ rotation per minute ဖြင့် လည်ရမယ့် speed rpms ကို သတ်မှတ်ပေးပါတယ်။ rpms ဟာ positive number ဖြစ်ရပါမယ်။

```
steppermotor.step(stepsRequired);
```

ကိုယ်လိုချင်တဲ့ step အရေအတွက်အတိုင်း motor ကို လည်စေမယ့် method ဖြစ်ပါတယ်။ speed ကတော့ သူ့ရဲ့ အပေါ်မှာ ကြိုတင်သတ်မှတ်ထားတဲ့ setSpeed () method ကို ကြည့်ပါတယ်။ ဒီ function ဟာ blocking function ဖြစ်တာကြောင့် သတ်မှတ်ထားတဲ့ step အရေအတွက်ကို motor က လည်လို့ မပီးသေးမချင်း code နောက်တစ်ကြောင်းကို run မှာမဟုတ်ပါဘူး။ သူရဲ့ syntax ကတော့

```
objectName.step(steps);
```

Parameter ထဲက steps ကတော့ မော်တာလည်ရမယ့် step အရေအတွက်ဖြစ်ပါတယ်။ step ကို positive integer ကိုသုံးရင် direction တစ်ခုနဲ့ပြီး negative integer ကို သုံးရင် အခြား direction တစ်ခုနဲ့ လည်မှာဖြစ်ပါတယ်။

```
steppermotor.setSpeed(1);
StepsRequired=4;
steppermotor.step(StepsRequired);
delay(2000);
```

Speed 1rpms နဲ့ 4 step လည်ပြီး 2 second ရပ်ပါမယ်။

```
StepsRequired=STEPS_PER_OUT_REV/2;
steppermotor.setSpeed(100);
steppermotor.step(StepsRequired);
delay(1000);
```

မော်တာ shaft ရဲ့ တစ်ဝက်တိတိလည်ချင်တာကြောင့် stepsRequired variable ထဲကို shaft အပြည့်လည်တဲ့ တန်ဖိုးကို 2 နဲ့စားပြီး ထည့်လိုက်ပါတယ်။ ဒါကြောင့် မော်တာဟာ speed 100 rpms နဲ့ တစ်ဝက်တိတိလည်မှာဖြစ်ပြီး 1 second ရပ်မှာဖြစ်ပါတယ်။

```
StepsRequired=-STEPS_PER_OUT_REV/2;
steppermotor.setSpeed(700);
steppermotor.step(StepsRequired);
delay(2000);
```

ကျန်တဲ့ တစ်ဝက်ကိုတော့ speed 700 rpms နဲ့ လည်ပြီး 2 second ရပ်မှာဖြစ်ပါတယ်။

I2C Connection With LCD (Liquid Crystal Display)

ယခု Nano Shield တွင် External device များနှင့် Data Communicate လုပ်နိုင်ရန် J3 တွင် I2C pin out တစ်ခု ပါဝင်ပါတယ်။ Example အနေနဲ့ LCD ကို I2C bus တပ်ပြီး ချိတ်ဆက် အသုံးပြုပုံကို Example sketch နှင့်တကွ ဖော်ပြထားပေးပါတယ်။

Note: LCD ကို I2C bus နှင့် အသုံးပြုရန်အတွက် Arduino IDE တွင် built in ပါဝင်သော LiquitCrystalLibrary ဖြင့် အသုံးပြု၍ မရပါ။ အောက်တွင် ပေးထားသော Git repo link တွင် Library Zip file ကို ရယူ၍ Arduino IDE တွင် Sketch>>>Include Library>>>Add. ZIP library တွင် Download လူပ်ခဲ့သော Zip file ရွေးချယ်၍ Library install လုပ်ပြီးလျှင် Arduino IDE ကို ပိတ်ပြီး ပြန်ဖွင့်ကာ Library ကို အသုံးပြုနိုင်ပြီ ဖြစ်ပါတယ်။
Link for New Liquid Crystal Library
https://github.com/ribasco/new-liquidcrystal
Example sketch တွင်ပါဝင်သော
```
const int i2c_addr = 0×27;
```

Code line သည် I2C bus ရဲ့ address ကို variable ပေးထားခြင်း ဖြစ်ပါတယ်။ 0×27 သည် မိမိအသုံးပြု သော I2C bus ပေါ်မူတည်၍ ပြောင်းလဲ နိုင်ပါတယ်။ ထို့ကြောင့် Example sketch ကို မ run ခင် Scanner sketch ကို အရင် run ပြီး မိမိ အသုံးပြုသော I2C bus ရဲ့ address ကို သိရှိနိုင်ပါတယ်။

Pin Description (In PCB – Not required external hardware)

SDA	**A4**
SCL	**A5**

I2C Scanner Sketch

```
//--- Begin code (copy from here)---

#include "I2CScanner.h"

I2CScanner scanner;

void setup()

{

        Serial.begin(9600);

        while (!Serial) {};

        scanner.Init();

}

void loop()

{

        scanner.Scan();

        delay(5000);

}
```

```
//--- End code (copy to here)---
```

I2C Scanner Code ရှင်းလင်းချက်

```
#include "I2CScanner.h"
```

I2C scanner library ကို အသုံးပြုဖို့ include လုပ် ကြေငြာပါတယ်။

```
I2CScanner scanner;
```

I2CScanner class ရဲ့ "scanner" လို့ ခေါ်တဲ့ object ကို creat လုပ်ပါတယ်။

```
Serial.begin(9600);
```

I2C device ရဲ့ address ကို serial monitor ကနေ ကြည့်မှာ ဖြစ်လို့ Serial monitor ကို baud rate 9600 နဲ့ begin စလုပ်ပါတယ်။

```
while(!Serial){}
```

```
scanner.Init();
```

Scanner initiate လူပ်ပါတယ်။ loop section ထဲမှာတော့

```
scanner.Scan();
```

```
delay(5000);
```

Scan ဖတ်ခြင်းကို 5စက္ကန့်တစ်ခါ လုပ်မာဖြစ်ပါတယ်။

Example Sketch

```
//--- Begin code (copy from here)---
```

```
#include <Wire.h>
```

```
#include <LiquidCrystal_I2C.h>
```

```
const int  en = 2, rw = 1, rs = 0, d4 = 4, d5 = 5, d6 = 6, d7 = 7, bl = 3;
```

```
const int i2c_addr = 0x27;// if u don't know the address, first i2c scanner sketch
```

```
LiquidCrystal_I2C lcd(i2c_addr, en, rw, rs, d4, d5, d6, d7, bl, POSITIVE);
```

```
void setup() {
lcd.begin(20,4);
  // put your setup code here, to run once:
  lcd.setCursor(0,0);
  lcd.print("Hello world ");
}
void loop() {
  // put your main code here, to run repeatedly:
 }
//--- End code (copy to here)---
```

LCD code ရှင်းလင်းချက်

```
#include<wire.h>
```

I2C device ကို အသုံးပြုပြီး communicate လုပ်မှာဖြစ်လို့ wire library ကို include ကြေငြာပေးရပါတယ်။

```
#include<LiquidCrystal_I2C.h>
```

LCD ကို I2C bus ချိတ်ပြီး သုံးမှာဖြစ်လို့ LiquidCrystal_I2C library ကို include လုပ်ပါတယ်။

```
const int en=2,rw=1,rs=0,d4=4,d5=5,d6=6,d7=7,bl=3;
```

LCD ရဲ့ pin number တွေကို variable တွေအနေနဲ့ သတ်မှတ်ပေးပါတယ်။

```
const int i2c_addr = 0x27;
```

မိမိ အသုံးပြုတဲ့ I2C device ရဲ့ address ကို variable ထားပါတယ်။

```
LiquidCrystal_I2C lcd(i2c_addr,en,rw,rs,d4,d5,d6,d7,bl,POSITIVE);
```

LiquidCrystal_I2C class ရဲ့ "lcd" ဆိုတဲ့ object လေးကို လိုအပ်တဲ့ parameter တွေနဲ့ creat လုပ်ပါတယ်။

setup ထဲမှာတော့

```
lcd.begin(20,4);
```

မိမိအသုံးပြုတဲ့ LCD အမျိုးအစားရဲ့ row အရေအတွက် colum အရေအတွက်ကို parameter အနေနဲ့ထည့်ပြီး lcd ကို စတင်ပါတယ်။

```
lcd.setCursor(0,0);
```

LCD မှာ စာတွေ စပေါ်ဖို့ဆိုရင် ပထမစာလုံးရဲ့ ပေါ်ရမယ့်နေရာကို setCursor() method ကို သုံးပြီး သတ်မှတ်ပေးရပါမယ်။ parameter ထဲကတော့ စပေါ်ရမယ့် cursor ချရမယ့် row နဲ့ colum ဖြစ်ပါတယ်။ အမြဲတမ်း "0" က စရေရပါမယ်။ (0,0) ဟာ အပေါ်ဆုံး row ရဲ့ ဘယ်ဘက်အစွန်းဆုံး colum ဖြစ်ပါတယ်။

```
lcd.print("Hello world");
```

ပေါ်ချင်တဲ့စာကို print() method သုံးပြီး ပေါ်ချင်တဲ့ string ကိုတော့ parameter အနေနဲ့ ထည့်ပေးရပါတယ်။

loop section ထဲမှာတော့ ဘာမှရေးစရာ မရှိပါ။

Software Serial UART Communication

Arduino Nano Shield နဲ့ တခြား device များ data communicate လုပ်နိုင်ဖို့ I2C အပြင် UART protocol ပါ အသုံးပြုနိုင်အောင် ထည့်ပေးထားပါတယ်။ UART ကိုမှ Software Serial ရော hardware Serial ပါ သုံးနိုင်အောင် Board မှာ Built in တည်ဆောက်ပေးထားပါတယ်။

Software Serial ကိုတော့ Arduino nano ရဲ့ digital pin 5 ကို SW_RX pin အဖြစ်လည်းကောင်း၊ digital pin 6 ကို SW_ TX အဖြစ်လည်းကောင်း၊ PCB လမ်းကြောင်းဖြင့် ချိတိဆက်ပေးထားပါတယ်။ External device နဲ့ ချိတ်ဆက်ဖို့ J4 မှာ power pin များဖြင့် ထုတ်ပေးထားပါတယ်။

Hardware Serial အနေနဲ့ သုံးမယ့် Arduino Nano ရဲ့ Digital pin o နဲ့ Digital pin 1 ကို j5 မှာ PCB လမ်းကြောင်းများဖြင့် တိုက်ရိုက် ချိတ်ဆက်ပီး TX RX pin, power pin များအဖြစ် ထုတ်ပေးထားပါတယ်။

ယခု Example မှာတော့ Software Serial ကို သူုံးပြီး Bluetooth Module လေးနဲ့ data communicate လုပ်ပုံလေးကို ဖော်ပြထားပါတယ်။ ယခု Example လေးကို ပြုလုပ်ဖို့ ဖုန်း App လေး တစ်ခု လိုအပ်ပါတယ်။ App ကို အောက်က Link တွင် Download ပြုလုပ်ပါ။

https://drive.google.com/file/d/1Utff-bzW3QQdjfxjnl0J2I-nvCewOAwD/view?usp=sharing

ဒီ app လေးကို အသုံးပြုပြီး ဖုန်းရဲ့ bluetooth ကနေတဆင့် Nano Shield နဲ့ ချိတ်ဆက်ထားတဲ့ bluetooth module ကို Data ပေးပို့မှာ ဖြစ်ပါတယ်။ Bluetooth module ကို ရောက်ရှိလာတဲ့ Data ကို arduino သို့ Software Serial (UART communication) pin တွေမှ တဆင့် ပေးပို့ပေးမှာ ဖြစ်ပါတယ်။ ရရှိလာတဲ့ Data ပေါ် မူတည်ပီး Nano Shield မှာ ပါဝင်တဲ့ LED D1 ကို on off control လုပ်သွားမှာ ဖြစ်ပါတယ်။

အောက်မှာ bluetooth module နဲ့ Arduino Nano Shield ရဲ့ ချိတိဆက်ပုံ pin description နှင့် Example sketch ကို ထည့်သွင်းပေးထားပါတယ်။

Pin Description (In PCB – Not required external hardware)
Software Serial (UART)

J4 Pinout	Arduino Nano Pinout
+5V Power Supply for External Device	+5V
GND for External Device GND Pin	GND
SW_RXD	D5
SW_TXD	D6

Hardware Serial (UART)

J5 Pinout	Arduino Nano Pinout
+5V Power Supply for External Device	+5V
GND for External Device GND Pin	GND
RX	D0
TX	D1

Connection between Arduino Nano Shield J4 and Bluetooth Module

J4	Bluetooth Module
+5V	+5V
GND	GND
SW_RX	TX
SW_TX	RX

Example Sketch

```
//--- Begin code (copy from here)---

#include <SoftwareSerial.h>
```

```
SoftwareSerial BT (5,6); //TX RX respectively
int ch ;
int led = 6;

void setup() {
  // put your setup code here, to run once:
  BT.begin(9600);
  Serial.begin(9600);
  pinMode(led,OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  if (BT.available()>0){
    ch = BT.read();
    Serial.println(ch);
    if(ch == 1){
      digitalWrite(led,HIGH);
    }

    if(ch == 0){
      digitalWrite(led,LOW);
    }
  }
```

```
}

//--- End code (copy to here)---
```

LED Control via Bluetooth (UART) code ရှင်းလင်းချက်

```
#include<SoftwareSerial.h>
```

UART protocol ကို သုံးပြီး communicate လုပ်မှာဖြစ်သလို Arduino မှာ hardware အရ တစ်ခါတည်းသတ်မှတ်ထားပေးပြီးသား Serial communication pin တွေကို သုံးမှာ မဟုတ်ပဲ မိမိ စိတ်ကြိုက် I/O pin ကို Sketch ထဲမှာ Serial communication pin အဖြစ် software အရ ပြောင်းလဲ သတ်မှတ် အသုံးပြုမှာ ဖြစ်လို့ SoftwareSerial library ကို include လူပ် ကြေငြာပါတယ်။

```
SoftwareSerial BT (5,6);
```

SoftwareSerial class ရဲ့ "BT" လို့ခေါ်တဲ့ object ကို creat လုပ်ပါတယ်။ bluetooth နဲ့ ချိတ်ဆက်အသုံးပြုမှာ ဖြစ်လို့ BT လို့ အလွယ်ပေးတာဖြစ်ပါတယ်။ parameter ကတော့ RX,TX pin အဖြစ် မိမိတို့စိတ်ကြိုက်ပြောင်းလဲ သတ်မှတ်ထားတဲ့ arduino pin number တွေကို ထည့်ပေးရမှာဖြစ်ပါတယ်။ သူ့ရဲ့ syntax ကတော့

```
SoftwareSerial objectName (RX,TX);
```

```
int ch;
```

Bluetooth ကနေ data အဖြစ်ပေးပို့လာမယ့် integer တန်ဖိုးကို store လူပ်ဖို့ variable declare လူပ်ခြင်းဖြစ်ပါတယ်။

```
int led = 6;
```

Led တပ်ဆင်ထားတဲ့ arduino pin number ကို variable အဖြစ် declare လုပ်ခြင်းဖြစ်ပါတယ်။

setup section ထဲမှာတော့

```
BT.begin(9600);
```

SoftwareSerial ကို baud rate (9600) နဲ့ data communicate လုပ်ဖို့ ready လုပ်တာဖြစ်ပါတယ်။

```
Serial.begin(9600);
```

Seial monitor ကနေ ကြည့်ဖို့ baudrate 9600 နဲ့ပဲ စပါတယ်။

```
pinMode(led,OUTPUT).;
```

Led pin ကို Mode သတ်မှတ်ပေးတာဖြစ်ပါတယ်။

```
BT.available() ;
```

Bluetooth ကနေ data ပို့လိုက်လို့ရှိရင် Arduino ထဲကို တိုက်ရိုက်တန်းမရောက်ပါဘူး။ serial buffer ထဲကို တစ် byte ချင်းရောက်တာဖြစ်ပါတယ်။ Serial buffer ထဲကို data ရောက်မရောက် စောင့်တာ ဖြစ်ပါတယ်။

```
if(BT.available()>0){

 //code

 }
```

Buffer ထဲမှာ Data ရောက်ခဲ့ရင် avaiable method ကို ခေါ်တဲ့အခါ 0 ထက်ကြီးမှာဖြစ်လို့ Data ရောက်ပီဆိုတာသိနိုင်ပါတယ်။ ဒါကြောင့် data ရောက်ခဲ့ရင် လူပ်ရမယ့် အလုပ်တွေကို ဒီ if ထဲမှာ ဆက်ရေးရမှာဖြစ်ပါတယ်။

```
ch=BT.read();
```

buffer ထဲက data ကို ဖတ်ပြီး ch ဆိုတဲ့ variable ထဲမှာ သွားသိမ်းတာ ဖြစ်ပါတယ်။

```
Serial.println(ch);
```

ရလာတဲ့ data မှန်မမှန် ကြည့်လို့ရအောင် serial monitor မှာ print လုပ်တာ ဖြစ်ပါတယ်။

```
if(ch == 1){

 digitalWrite(led,HIGH);

 }
```

data တန်ဖိုးက 1 ဆိုရင် မီးလင်းမှာဖြစ်ပါတယ်။

```
if(ch==0){

  digitalWrite(led,LOW);

}
```

data တန်ဖိုး 0 ဆိုရင် မီးပြန်ပိတ်မှာဖြစ်ပါတယ်။

Final Project

နောက်ဆုံးအနေနဲ့ Nano shield လေးကို သုံးပြီး project လေးတစ်ခု Example အနေနဲ့ ထည့်ပေးထားပါတယ်။ ဒီ project မှာတော့ DHT11 Sensor ကနေ Temperature နဲ့ Humidity တန်ဖိုးကို ရယူပြီး LCD screen မှာ ပြသပေးမှာ ဖြစ်ပါတယ်။ ဒါ့အပြင် Temperature 70°C ထက် ကျော်ပါက Buzzer မှာ Alarm မည်အောင်ရေးထားပေးပါတယ်။ စမ်းမယ်ဆိုရင်တော့ လိုအပ်သလို Temperature တန်ဖိုးကို program တန်ဖိုးကို အတိုးအလျော့ လုပ်ပီး စမ်းနိုင်ပါတယ်။

ဒါအပြင် Pushbutton deboung ကို Relay နဲ့ တွဲသုံးပေးထားပါတယ်။ Relay မှာ မီးသီး တစ်လုံး ချိတ်ဆက်ပြီး Button ကို တစ်ချက်နှိပ်လျှင် မီးလင်း နောက်တစ်ချက် ထပ်နှိပ်လျှင် မီးပြန်ပေးအောင် ရေးပေးထားပါတယ်။

Example Sketch

```
//--- Begin code (copy from here)---

#include <Wire.h>

#include <LiquidCrystal_I2C.h>

const int  en = 2, rw = 1, rs = 0, d4 = 4, d5 = 5, d6 = 6, d7 = 7, bl = 3;

const int i2c_addr = 0x27;

LiquidCrystal_I2C lcd(i2c_addr, en, rw, rs, d4, d5, d6, d7, bl, POSITIVE);


#include "DHT.h";

#define DHTpin 7

#define DHTTYPE DHT11

DHT dht(DHTpin,DHTTYPE);

float hum; //store humidity value in percent

float temp; //store temperature value in Celcius


int buzzerPin = 5;

int LED1 = 6;

int LED2 =7;

int buttonPin = 8;

int relayPin = A7;


int buttonState;
```

```
int relayState = HIGH;
int lastButtonState = LOW;

unsigned long lastDebounceTime = 0;
unsigned long debounceDelay = 50;

void setup() {
  // put your setup code here, to run once:
  lcd.begin(16,2);
  dht.begin();
  pinMode(buzzerPin,OUTPUT);
  pinMode(LED1,OUTPUT);
  pinMode(LED2,OUTPUT);
  pinMode(buttonPin,INPUT);
  pinMode(relayPin,OUTPUT);

  digitalWrite(relayPin,relayState);

}

void loop() {
  // put your main code here, to run repeatedly:
  int reading = digitalRead(buttonPin);
  hum = dht.readHumidity();
```

```
temp = dht.readTemperature();
lcd.setCursor(0,0);
lcd.print("Temp = ");
lcd.print(temp);
lcd.print(" C");
lcd.setCursor(0,1);
lcd.print("humidity = ");
lcd.print(hum);
lcd.print("%");
if(temp >= 70){
  digitalWrite(LED1,HIGH);
  tone(buzzerPin,450);
  delay(500);
  digitalWrite(LED1,LOW);
  digitalWrite(LED2,HIGH);
  noTone(buzzerPin);
  delay(500);
  digitalWrite(LED2,LOW);
}

if (reading != lastButtonState) {
  // reset the debouncing timer
  lastDebounceTime = millis();
}
```

```
  if ((millis() - lastDebounceTime) > debounceDelay) {
    // whatever the reading is at, it's been there for longer
    // than the debounce delay, so take it as the actual current state:

    // if the button state has changed:
    if (reading != buttonState) {
      buttonState = reading;

      // only toggle the LED if the new button state is HIGH
      if (buttonState == HIGH) {
        relayState = !relayState;
      }
    }
  }

  // set the LED:
  digitalWrite(relayPin, relayState);

  // save the reading.  Next time through the loop,
  // it'll be the lastButtonState:
  lastButtonState = reading;

}//--- End code (copy to here)---
```