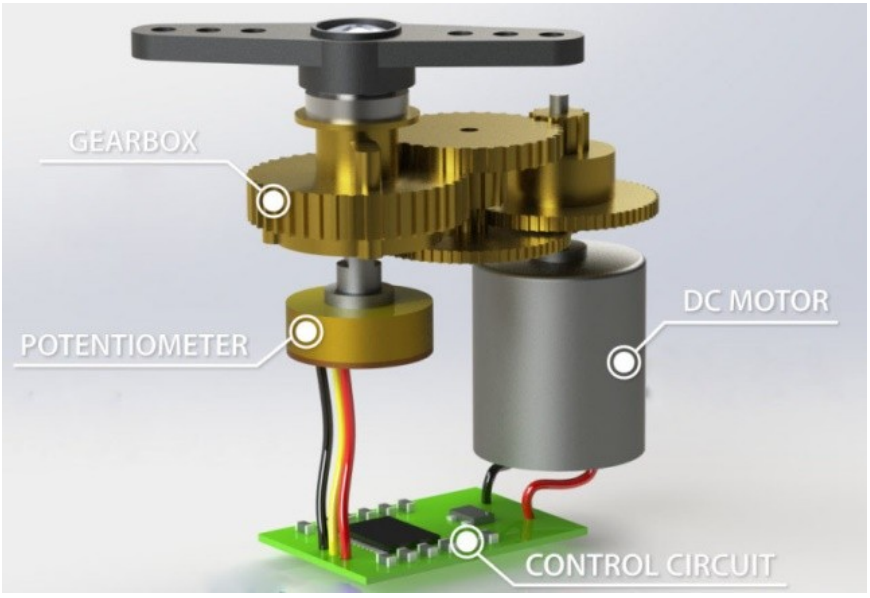
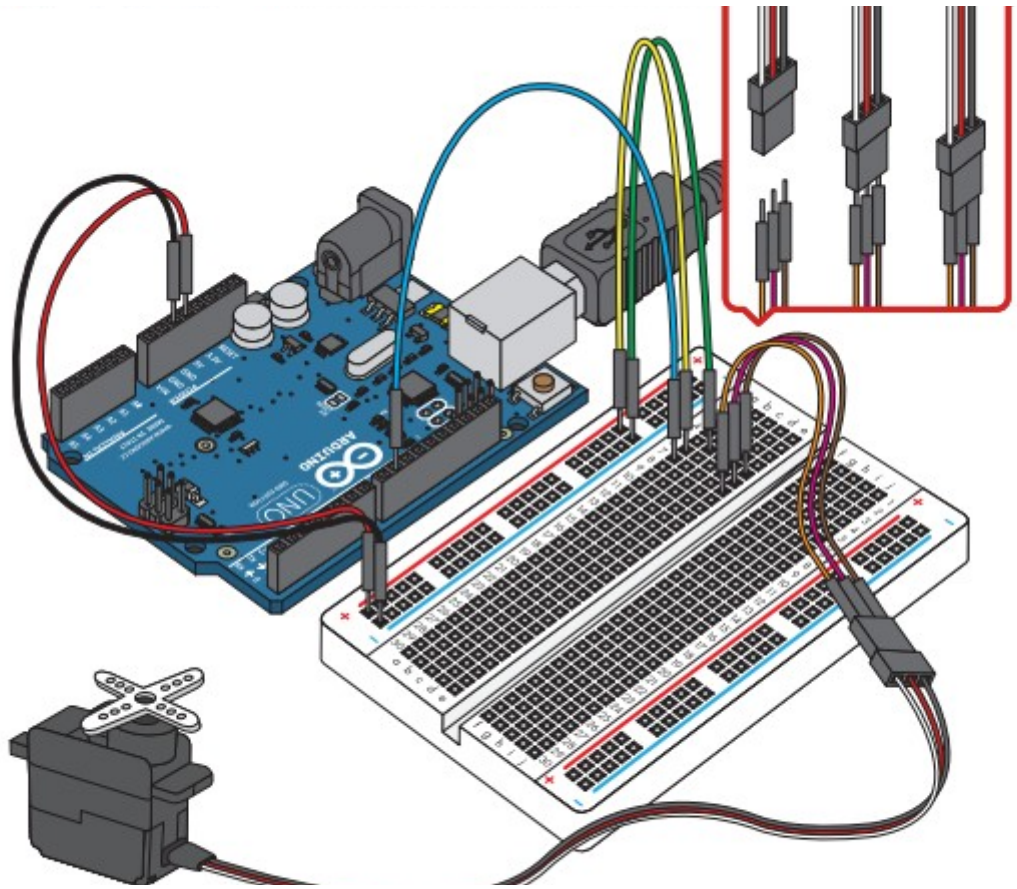
1.2 СЕРВО ҚОЗҒАЛТҚЫШ. ҚАДАМДЫҚ МОТОР

Сервомотор - бұрыштық немесе сызықтық позицияны, жылдамдықты және үдеуді дәл басқаруды қамтамасыз ететін айналмалы немесе сызықты жетек. Сервомоторлар жоғары айналу моментін қамтамасыз ету үшін позициялық сенсорлармен және редуктормен біріктірілген. Сервомоторлар белгілі бір қозғалтқыш класы емес, бірақ сервомотор термині жабық контурды басқару жүйесінде пайдалану үшін қолайлы қозғалтқышқа сілтеме жасау үшін жиі қолданылады. Сервомоторлар робототехникада, автоматтандырылған өндірісте және CNC машиналарында керемет қолданбасын тапты. Сервомоторлардың негізгі пайдалы қасиеттерінің бірі - позициялау бойынша кері байланысты қамтамасыз ететін қосылған потенциометрі бар кірістірілген басқару тақтасы (6.2.1 сурет).



Сурет-6.2.1. Серво қозғалқышының жүйесі



Сурет-6.2.2. Серво қозғалтқышының ардуиномен қосылу схемасы

6.2.2-суреттегі стандартты сервомоторларда қозғалтқыштың үш сымы бар, қара сым жерге тұйықталған, қызыл сым қуат сымы, сигнал сымы сары; кейде ақ түсті болуы мүмкін. Кейбір жағдайларда сервомотордың төрт сымы болады; қосымша сым кері байланысты қамтамасыз етеді. Мұндай кері байланыс роботтар мен мехатрондық жүйелер үшін оқытуды басқару әдістерін қолдануға мүмкіндік береді.

Сонымен қатар, сервомоторлардың негізгі артықшылықтарының бірі - қозғалтқыштың кірістірілген қозғалтқыш тақтасы. Сондықтан сервомоторлар қосымша қозғалтқыш жетектерін қажет етпейді және оларды микроконтроллерге тікелей қосуға болады. Сонымен қатар, сервомоторларды басқару бағдарламасының коды басқа тұрақты ток қозғалтқыштарына қарағанда қарапайым.

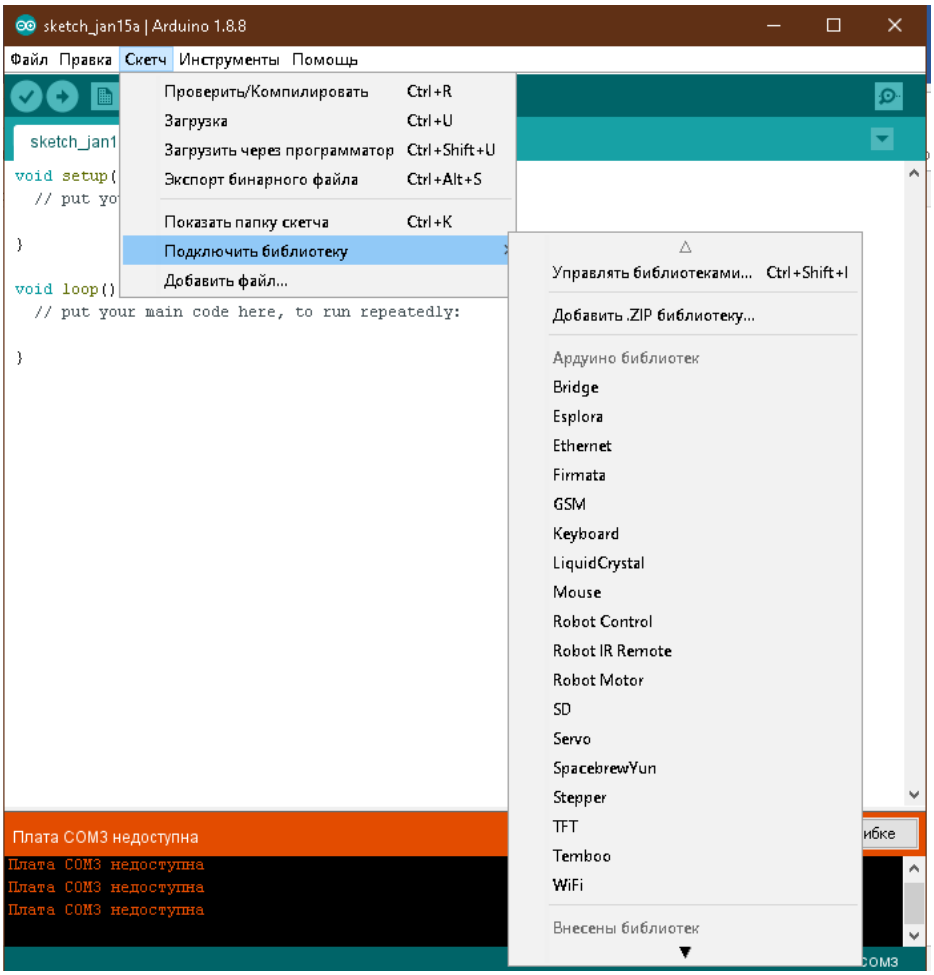
Кітапханалардың көмегімен өз мүмкіндіктеріңізді кеңейтіңіз:

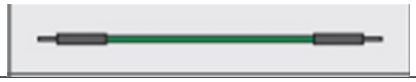
Arduino-да орындау үшін пайдалы командалар жиынтығы бар, олар негізгі енгізу / шығару операциялары. Командалар логиканы қолдана отырып шешім қабылдай алады, сонымен қатар математикалық есептерді шеше алады. Бірақ Arduino-ның күші бұл қызығушылық танытатын адамдардың үлкен қауымдастығы, сонымен қатар олардың жұмысының нәтижелерімен бөлісуге дайын болуы болып табылады.

Кітапханалар-бұл жаңа командалардың жинақтары, оларды сіздің жобаңызға оңай қосуға болады. Ардуинода **Wire, LiquidCrystal, Ethernet, Servo** және т. б. осы сияқты бірнеше пайдалы кітапханалар бар. Servo-сервожетектердің кітапханасын біз осы тәжірибемізде қолданамыз. Қосымша ақпаратты осы жерден қараңыз:

http: //arduino.cc/en/Reference/Libraries

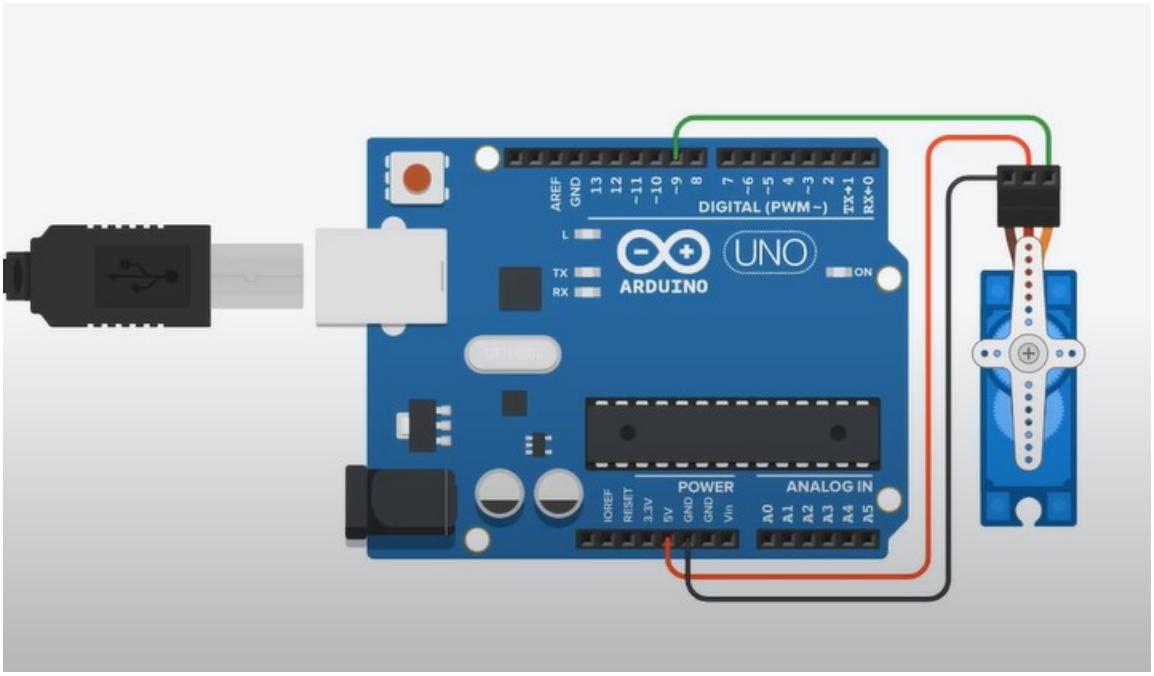
Егер сіз жаңа датчикті немесе құрылғыны қолданғыңыз келсе, онда өзара әрекеттесу үшін кітапханаларды жүктеп алуыңыз керек. Көптеген датчиктер үшін кітапханалар жасалып қойылған. Сізге кітаханаларды жүктеу үшін Google және Yandex көмектеседі. Жобада кітапхананы пайдалану үшін оны Arduino IDE > Sketch > Import Library (Scetch> кітапхананы импорттау) мәзірінен таңдаңыз.



Сервоқозғалтқыш	
Сым	
Сым	
Сым	

Қажетті құрал-жабдықтар:

Берілген схеманы жинаңыз



Бағдарламасы/Код

Келесі кодты көшіріп, оны Arduino тақтасына жүктеңіз. Код жақсы түсіндірілген, сондықтан сіз оның қалай жұмыс істейтінін оңай түсініп, оны өз жобаларыңызға қосу үшін өзгерте аласыз.

```
#include <Servo.h>
int pos = 0;
Servo servo_9;
void setup()
{
  servo_9.attach(9, 500, 2500);
}
void loop()
{

  for (pos = 0; pos <= 180; pos += 1) {
  servo_9.write(pos);
    delay(15);
  }
 for (pos = 180; pos >= 0; pos -= 1) {
    servo_9.write(pos);
    delay(15);
  }
}
```

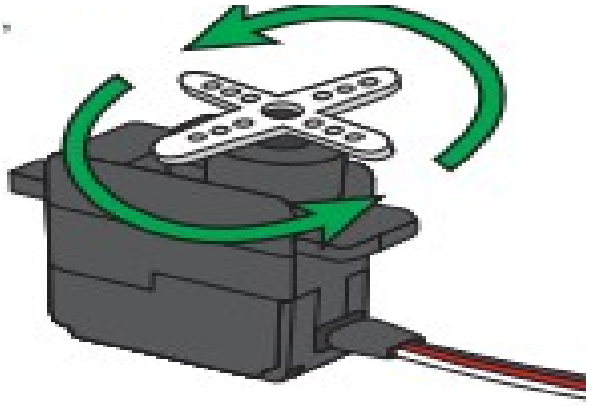
```
#include <Servo.h>
int pos = 0;
void setup()
{
pinMode(9,OUTPUT);
}
void loop()
{

while(pos<2500);

{

pos++;
digitalWrite(9, 1);
delayMicroseconds(pos);
digitalWrite(9, 0);
delayMicroseconds(2000);
  }
 }
```

БАҒДАРЛАМАҒА ШОЛУ

#include <Servo.h>	# include сіздің жобаңызға кітапхананы (немесе кез-келген басқа файлды) енгізетін арнайы процессор директивасы. Сіз бұл команданы қолмен енгізе аласыз немесе орнатылған кітапхананы Arduino IDE мәзірінен таңдай аласыз: Скетч> Импортировать Библиотеку> Add Library
Servo servo1; servo1.attach(9);	Servo кітапханасында әртүрлі сервожетектерін басқаруға мүмкіндік беретін жаңа командалар бар. Arduino Сервоқозғалтқышын басқаруға дайындау үшін алдымен әр серво үшін Servo "нысанын" құру керек (біз оны "servo1" деп атадық), содан кейін оны сандық портқа (attach) бекіту керек (бізде 9 порт бар).
servo1.write(180);	Сервожетектер шеңберде айнала алмайды, бірақ белгілі бір позицияны ала алады. Servo кітапханасының write() командасын сервожетегін белгіленген градус санына (0-ден 180-ге дейін) орнату үшін қолданамыз. Есіңізде болсын, сервожетекке белгілі бір позицияға жету үшін уақытты қажет етеді, сондықтан delay() командасын қолданамыз.

Сіз не көруіңіз керек:

Сіз сервоқозғалтқыштың әр түрлі позициялар әр түрлі жылдамдықпен ауысатынын көруіңіз керек. Егер қозғалтқыш қозғалмаса, порттардың жалғануын тексеріп, кодтың Arduino-да жүктелгеніне көз жеткізіңіз немесе төмендегі ақаулықтарды жою бойынша кеңестерді қолданыңыз.



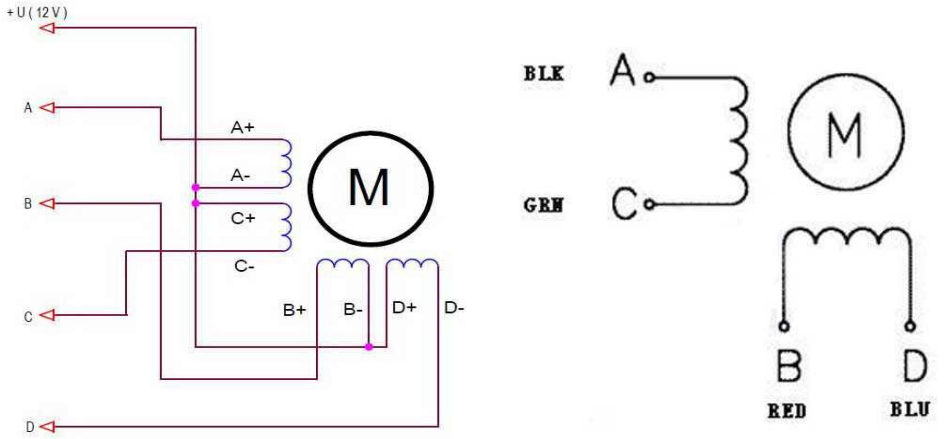
Мүмкін болатын қиындықтар:	**Мұндай схемаларды өмірде қалай қолдануға болады?**
Servo айналмайды Түрлі-түсті сымдармен бірге сервоны кері бағытта қосуға мүмкіндік бар. Порттардың жалғануының дұрыстығын тексеріңіз.	Автомобиль зауыттарында қолданылатын роботты механикалық қолдар.
Жұмыс істемейді Кең таралған қателік бірі +5 вольт және жер сымдарын қосуды ұмытып кету.	
Серпіліп қозғалады Егер серво серпіліспен қозғала бастаса және Arduino бортындағы жарық диоды жанса, бұл қуат жеткіліксіз екенін білдіреді. USB орнына желілік адаптерді пайдаланыңыз, бұл мәселені шешуі керек.	

Қадамдық қозғалтқыш, сонымен қатар қадамдық немесе қадамдық қозғалтқыш ретінде белгілі, щеткасыз тұрақты ток электр қозғалтқышы, ол толық 360 градус айналуды шағын қадамдарға бөледі. Қозғалтқыштың орнына қозғалыс бағытын өзгертуге де бұйрық (команда) беруге болады. Қадамдық қозғалтқыштар кері байланыс үшін позиция сенсорларымен жабдықталуы мүмкін. Сонымен қатар, қадамдық қозғалтқыштар қажетті бұрышты және шығыс моментін жақсы ұстай алады, бұл қадамдық қозғалтқыштарды робототехника мен CNC машиналарында кеңінен қолдануға мүмкіндік береді. Барлық дерлік CNC машиналары, соның ішінде 3D принтерлері, соңғы эффектор құралын басқару үшін қадамдық қозғалтқыштарды пайдаланады. Басқа тұрақты ток қозғалтқыштарымен салыстырғанда қадамдық қозғалтқыштар жұмыстың дәлдігін қамтамасыз ете алады.

Орамдардың негізінде қадамдық қозғалтқыштар екі негізгі түрге бөлінеді: униполярлы және биполярлы.

Униполярлы қадамдық қозғалтқыштың бір фазада орталық шүмегі бар бір орамасы бар; фазада үш сым және екі фазалы қозғалтқыш үшін алты сым болуы керек. Кейде қуат көзінің орамдары бір-біріне байланады. Ток орталық шүмектен орамның соңына бір бағытта ағып кетеді. Бірақ орамның

екінші ұшын жерге қосу қарама-қарсы бағытты тудырады. Осылайша, бірполярлы қозғалтқыштың қозғалтқышы деген атау тікелей алға қойылған. MOSFET транзисторларын пайдалану арқылы бірполярлы қозғалтқышты басқару өте қарапайым. Дегенмен, униполярлы қадамдық қозғалтқыш биполярлы қозғалтқыштар сияқты жоғары айналу моментін қамтамасыз ете алмайды, бірақ униполярлы қозғалтқыштар жоғары жылдамдықты талап ететін қолданыстарда өте қолайлы.



А) Униполярлы қадамдық мотор Б) Биполяр қадамдық мотор

Сурет-6.3.1. Униполяр және биполяр қадамдық қозғалтқыштың құрылымдық схемасы.

Биполярлы қозғалтқыштар фазада бір ғана орам болатындай етіп жасалған. Стандартты құрылым бір фазаға екі өткізгіш және екі фазалық биполярлы қозғалтқыш үшін төрт сым болып табылады. Биполярлық атау токтың екі бағытта, алға және кері бағытта ағуы дегенді білдіреді (6.3.1-суретті қараңыз). Биполярлы қозғалтқыштарды жүргізу үшін Н-көпір тізбегін ескеру қажет. Биполярлық қозғалтқышты басқару кодтауы кезінде екі негізгі ерекшелікті ескеру керек, бағыт және қадам. Униполярлы қадамдық қозғалтқышымен салыстырсаңыз, биполярлы қадам қозғалтқыштары күрделірек. Робототехникада биполярлы қадамдық қозғалтқыштар олардың жоғары айналу күші мен дәлдігіне байланысты көбірек қолданылады.

Қадамдық қозғалтқыштардың үш негізгі жұмыс режимі бар. Негізгі мысал ретінде 200 тістері бар роторды пайдалану жұмыс режимдері:

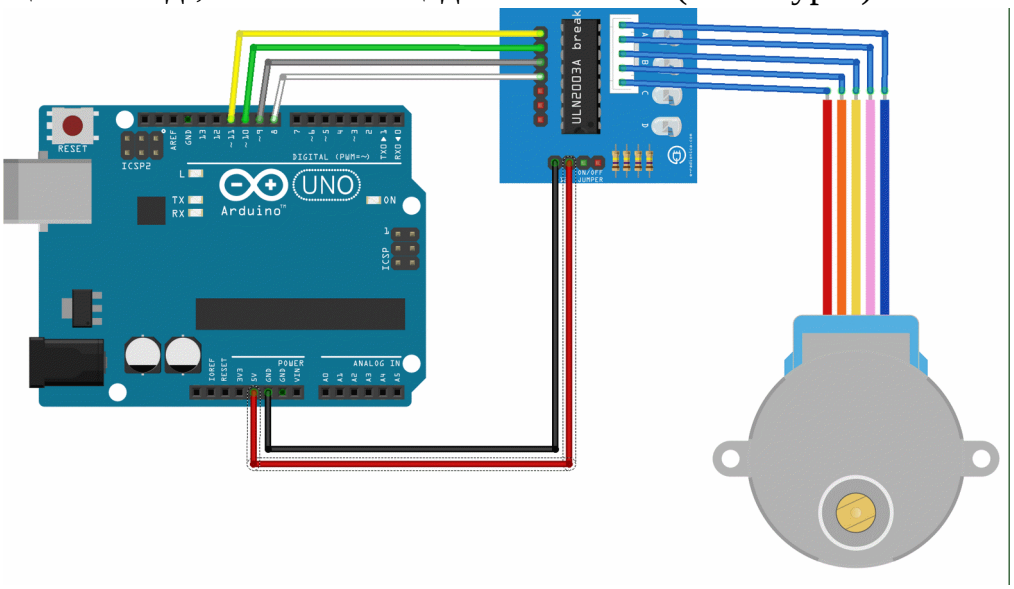
1. Толық қадам режимі: Ротордың айналуының әрбір қадамы 1,8 градусқа тең болғанда, біліктің толық айналуы 200 қадамды қажет етеді.

2. Жартылай қадам режимі: Ротордың айналуының әрбір қадамы 0,9 градусқа тең болғанда, біліктің толық айналуы 400 қадамды қажет етеді.

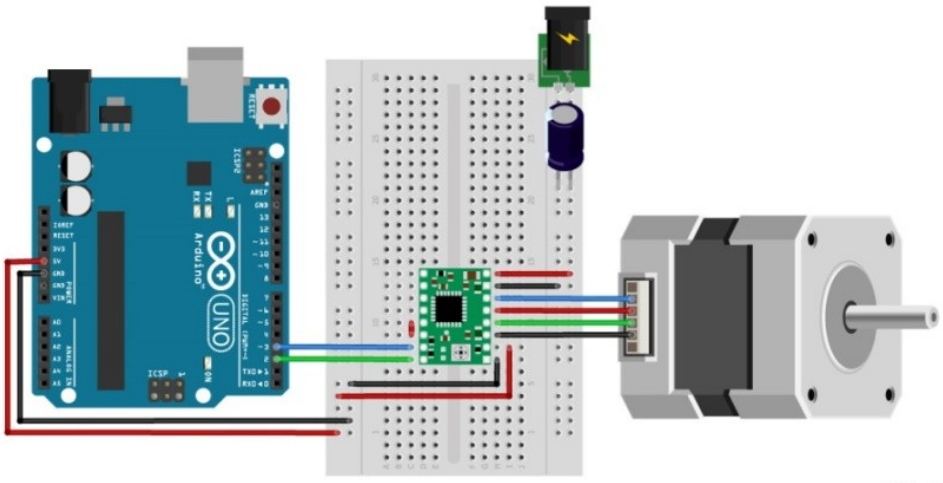
3. Микро қадам режимі: 360 градус айналу үшін қозғалтқыш 51 200 қадамнан өтуі керек; әрбір қадам 0,007 градусқа тең.

Қадамдық қозғалтқыштарды басқару жүйелері

Робототехника мен инженерияда қадамдық қозғалтқыштарды басқару роботты әзірлеу және жобалау үшін өте маңызды. Бірполярлы және биполярлы қозғалтқыштарды басқару әдістері мен программасы әртүрлі. Біріншіден, негізгі айырмашылық - мотор драйвері; мысалы, мотор драйвері ULN2003A плата арқылы униполярлы қадамдық қозғалтқыш жетектері. Бұл нарықта коммерциялық қол жетімді кең таралған мотор драйверлерінің бірі. Сонымен қатар, биполярлы қозғалтқыштарда Trinamic Motion, A4988, DRV8825 және «easydriver» тақтасының TMC платасы сияқты мотор драйверлерінің көптеген түрлері бар. Екіншіден, униполярлы және биполярлы қозғалтқыштарды кодтау. Униполярлы қадамдық қозғалтқыш (6.3.2-сурет) контроллерге 4 пиндік сымды қосуды және барлық 4 пиндері пайдалану арқылы кодты жасауды қажет етеді (1-кесте). Сондықтан униполярлы қадамдық қозғалтқышты басқаруға арналған басқару коды ұзақ және күрделі болар еді. Басқару биполярлы қозғалтқыштары тек екі түйреуішті қажет етеді; бағыт және қадам саны ғана (6.3.3-сурет).



Сурет-6.3.2. Униполярлы қадамдық қозғалтқыштың қосылу схемасы



Сурет-6.3.3. Биполярлық қадамдық қозғалтқыштың қосылу схемасы

Қадамдық қозғалтқыштардың басқару программасы

Бұл параграфта мысал код ретінде 1-кесте сағат тілімен және сағат тіліне қарсы 360 градусқа айналдыру программасы.

Кесте-1 Униполярлы мен биполярлы қадамдық қозғалтқыштарды басқару коды

Униполярлық қадамдық қозғалтқыш

```
#define IN1  11
#define IN2  10
#define IN3  9
#define IN4  8
int Steps = 0;
int Direction = 0;
int number_steps=50; //= 200/4
void setup() {
Serial.begin(9600);
pinMode(IN1, OUTPUT);
pinMode(IN2, OUTPUT);
pinMode(IN3, OUTPUT);
pinMode(IN4, OUTPUT);
}
void loop()
{
//1 rotation counter clockwise
stepper4(number_steps);
delay(1000);
//1 rotation clockwise
stepper4(-number_steps);
delay(1000);
//Keep track of step number
for(int thisStep=0;thisStep<number_steps;thisStep+
+){
stepper4(1);
}
delay(1000);
for(int thisStep=number_steps;thisStep>0;thisStep--)
{
stepper4(-1);
}
delay(1000);
}
void stepper4(double nbStep){
if(nbStep>=0){
Direction=1;
}else{
Direction=0;
nbStep=-nbStep;
}
for (int x=0;x<nbStep*4;x++){
switch (Steps) {
case 0:  // 1010
digitalWrite(IN1, HIGH);
digitalWrite(IN2, LOW);
digitalWrite(IN3, HIGH);
digitalWrite(IN4, LOW);
break;
```

Биполярлық қадамдық қозғалтқыш

```
#define dirPin 2
#define stepPin 3
#define stepsPerRevolution 200
void setup() {
// Declare pins as output:
pinMode(stepPin, OUTPUT);
pinMode(dirPin, OUTPUT);
}
void loop() {
// Set the spinning direction clockwise:
digitalWrite(dirPin, HIGH);
// Spin the stepper motor 1 revolution slowly:
for (int i = 0; i < stepsPerRevolution; i++) {
// These four lines result in 1 step:
digitalWrite(stepPin, HIGH);
delayMicroseconds(2000);
digitalWrite(stepPin, LOW);
delayMicroseconds(2000);
}
delay(1000);
// Set the spinning direction counterclockwise:
digitalWrite(dirPin, LOW);
// Spin the stepper motor 1 revolution quickly:
for (int i = 0; i < stepsPerRevolution; i++) {
// These four lines result in 1 step:
digitalWrite(stepPin, HIGH);
delayMicroseconds(1000);
digitalWrite(stepPin, LOW);
delayMicroseconds(1000);
}
```

```
case 1:  // 0110
digitalWrite(IN1, LOW);
digitalWrite(IN2, HIGH);
digitalWrite(IN3, HIGH);
digitalWrite(IN4, LOW);
break;
case 2:  //0101
digitalWrite(IN1, LOW);
digitalWrite(IN2, HIGH);
digitalWrite(IN3, LOW);
digitalWrite(IN4, HIGH);
break;
case 3:  //1001
digitalWrite(IN1, HIGH);
digitalWrite(IN2, LOW);
digitalWrite(IN3, LOW);
digitalWrite(IN4, HIGH);
break;
}
delayMicroseconds(2000);
if(Direction==1){ Steps++;}
if(Direction==0){ Steps--; }
if(Steps>3){Steps=0;}
if(Steps<0){Steps=3; }
}
}
```