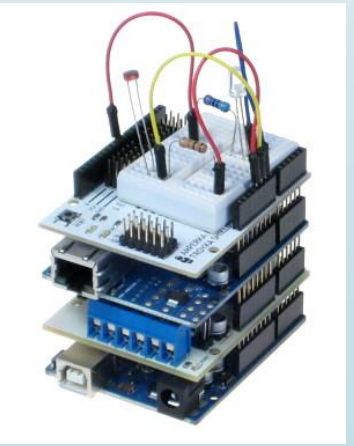
Цифрлық контактілерден деректерді оқу



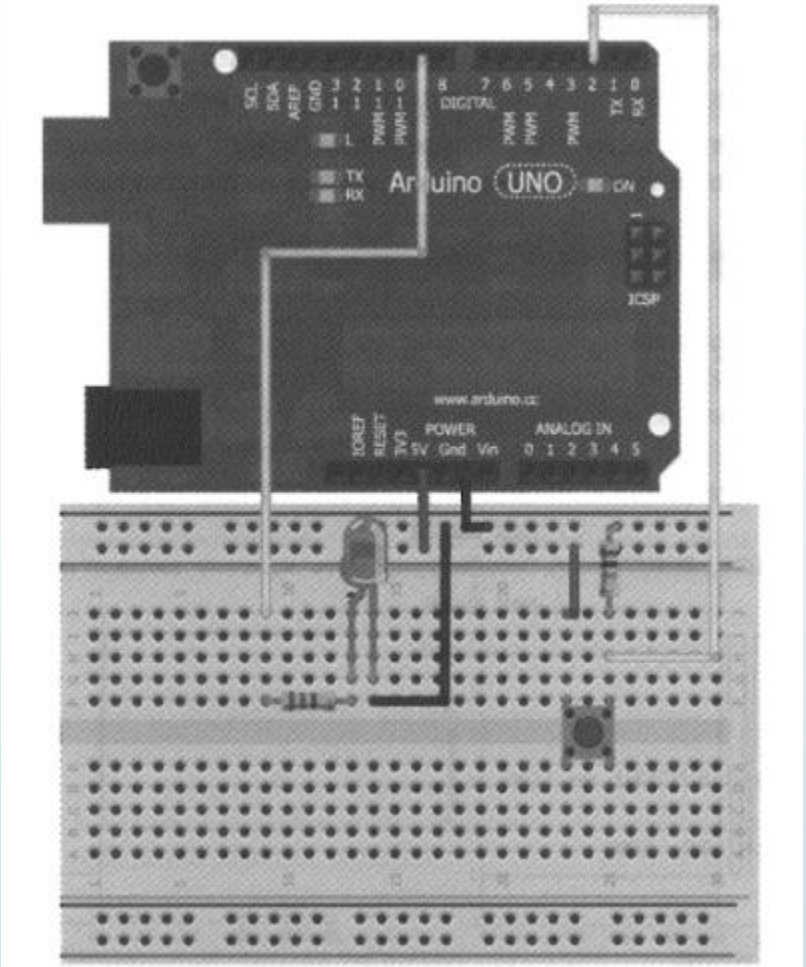
Цифрлық контактілердің тағы бір функциясын қарастырайық. Осы уақытқа дейін оларды цифрлық сигнал және ЕИМ сигналын генерациялап, шығыс ретінде қолдандық.

Келесі қадам - Arduino платасының контактілерінің кіріс ретінде жұмыс істеуі. Бұл, мысалы, нақты уақыт режимінде құрылғымен өзара әрекеттесу үшін ажыратып-қосқыштар мен батырмаларды қосуға мүмкіндік береді.

Тартушы резисторы бар цифрлық кірістерді оқу

4-дәріс 2-суретте көрсетілген схеманы өзгертейік. Цифрлық контактіге батырманы және тарткыш резисторды қосамыз, нәтижесінде схема 1 суретте көрсетілгендей болады.

Тартушы резистор - электр сигналы таралатын өткізгіш пен қоректендіруші (pull - up resistor - жоғарыға тартушы/қоректендіруге тартушы резистор) арасында немесе өткізгіш пен жер арасында қосылған резистор (pull - down resistor-тартушы резистор).



Сурет 1. Arduino платасына батырма және жарықдиодты қосу

Батырманың күйін сұрауға арналған бағдарламаны жазудан бұрын, осы схемадағы резистордың мақсатын түсіну маңызды. Барлық цифрлық кірістер үшін кіріс контактісінде «үнсіз келісім бойынша» мәнді орнату үшін қосымша жоғарыға тартушы (pull-down) немесе төменге тартушы (pull-up) резисторды қажет етеді. 1 суреттегі схемада 10 кОм резистор жоқ деп ойлаңыз. Мұндай жағдайда батырманы басқан кезде шығыс HIGH мәнін қабылдайды. Бірақ, батырма басылмаса не болады? Бұл жағдайда кіріс ешнәрсемен байланыстырылмаған, басқаша айтқанда «ауада тұрады».

Шығыс 0V немесе 5V кернеуіне физикалық түрде қосылмағандықтан, оның мәнін оқу күтпеген нәтижелер беруі мүмкін. Жақын орналасқан шығыстардағы электрлік кедергілер кернеудің мәні HIGH және LOW арасында ауытқып отыруына әкелуі мүмкін. Оны болдырмас үшін, тартушы резисторды 1 суреттегідей қосады.

Батырма басылмаған кезде және кіріс контактісі 10кОм тартушы резистор арқылы жерге қосылғанда не болатынын қарастырайық. Резистор арқылы жылыстау тогы өтеді және кіріс контактісінде кернеудің мәні LOW болады. 10 кОм - тартушы резистор үшін кең таралған мән. Батырманы басқанда кіріс контактісі 5 В шинаға тікелей қосылады. Ток енді екі жолмен жүруі мүмкін:

♦ 5 В шинасына басылған батырманың нөлдік кедергісі арқылы;

♦ резистордың жерге жоғары кедергісі арқылы.

Ом заңына сәйкес ток әрқашан ең аз кедергі жолымен жүреді. Токтың көп бөлігі тұйықталған батырма арқылы өтеді және кірісте HIGH мәні орнатылады.

Тартушы резисторлар маңызды, себебі, олар батырманы басқан кезде, 5В және жер арасында батырма қысқа тұйықталуды тудырмайтынына және кіріс контактісі «ауада бос» қалмауына кепілдік береді.
Енді қарастырып отырған схемаға бағдарлама жазайық. Батырма басылған кезде жарықдиоды жануы керек, ал батырма босатылған кезде сөнуі керек (1 листинг).

Листинг 1. Батырманың көмегімен жарықдиодын қосу

```
const int LED=9;      //  9 контакт жарықдиодын қосу үшін
const int BUTTON=2;   //  2 контакт батырманы қосу үшін

void setup()
{
  pinMode (LED, OUTPUT);    //  Жарықдиодының контактісін
                            //  шығыс ретінде орнату
  pinMode (BUTTON, INPUT);  //  Батырманың контактісін
                            //  кіріс ретінде орнату
}

void loop()
{
  if (digitalRead(BUTTON) == LOW)
  {
    digitalWrite(LED, LOW);
  }
  else
  {
    digitalWrite(LED, HIGH);
  }
}
```

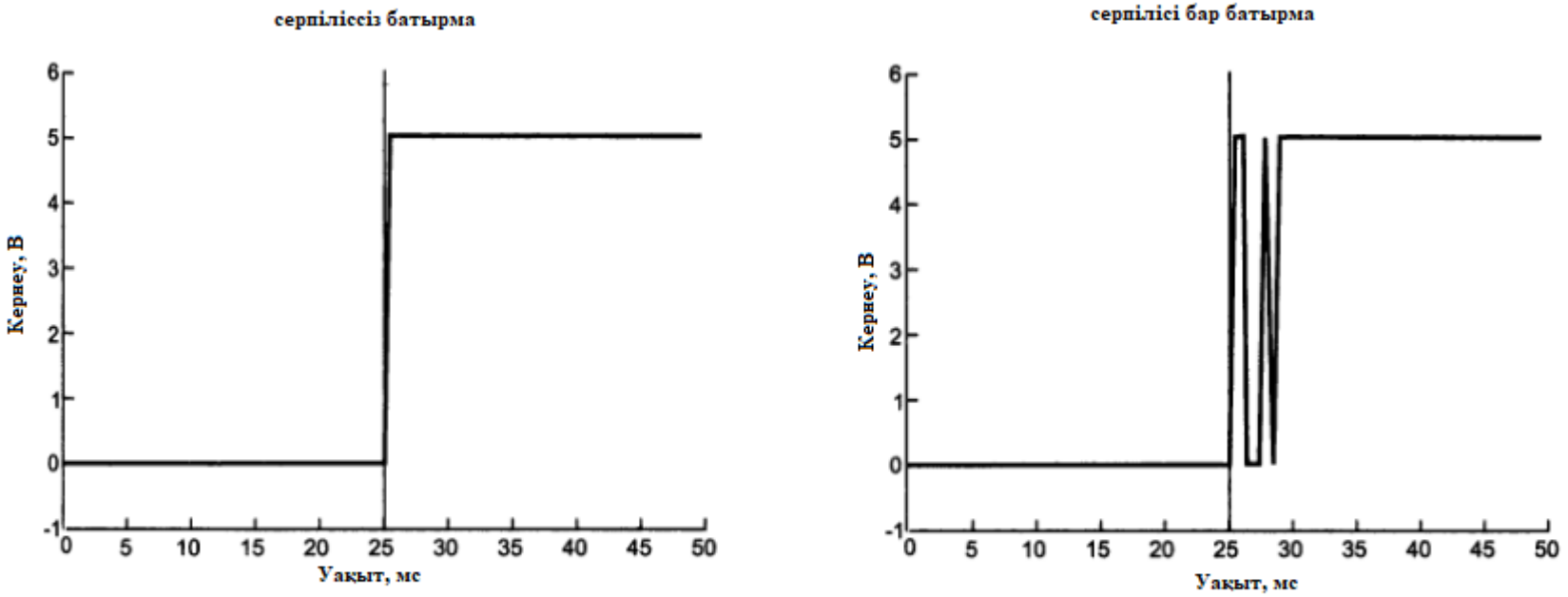
1 листингте кейбір жаңа элементтер енгізілді: digitalRead () функциясы және if / else операторы. Батырманың контактісі үшін int типті BUTTON тұрақтысы қосылды. Сонымен қатар, setup () функциясында BUTTON контактісін кіріс ретінде орнатамыз. Бұл міндетті емес, себебі Arduino шығыстары үнсіз келісім бойынша кірістер болып табылады. digitalRead функциясы кірістегі сигналдың мәнін оқиды. Батырма басылса, digitalRead () HIGH (лог. 1) мәнін қабылдайды. Егер батырма басылмаса, онда LOW (лог. 0) қабылданады.

if () операторының ішіндегі мазмұнды тексерейік.
if операторының ішіндегі шарт ақиқат болса (батырма басылмаса, digitalRead () == LOW), digitalWrite (LED, LOW) функциясы шақырылады (диод сөнеді). Басқа жағдайда, (батырма басылғанда) else операторынан кейінгі код орындалады (жарықдиоды digitalWrite (LED, HIGH) функциясымен қосылады).

Батырманың серпілуін (дребезг) жою

Жарықдиодты жарықтандыру үшін батырманы үнемі басып тұру ыңғайлы ма? Жарықдиодты қосу үшін түймені бір рет басып, сөндіру үшін оны қайтадан басу әлдеқайда жақсы. Осындай нұсқамен жарықдиодты жағу үшін батырманы басыңыз, ұстап тұрудың қажеті жоқ. Өкінішке орай, бұл айтылғандай оңай емес. Кірістегі сигналдың мәнін жай ғана оқи алмайсыз, контактінің серпілуі деп аталатын құбылысты ескеру қажет.

Кәдімгі батырмалар серіппелі механикалық құрылғылар болып табылады. Батырманы басқан кезде сигнал тек төменнен жоғарыға өзгеріп ғана қоймайды, сондай ақ, ол LOW деңгейі орнатылмай тұрып, бірнеше миллисекунд бойы өз мәнін қайта-қайта өзгертеді. Күтілетін процесс пен нақты процестің арасындағы айырмашылықты, батырманың сигналының осциллограммалары арқылы алынған 2 сурет сипаттайды.



Сурет 2. Батырманың серпілу эффектісі

Батырма физикалық түрде 25 мс басылды. Батырманың күйін контакт кірісіндегі мәнді оқу арқылы анықтауға болады деген болжам (сол жақтағы график) дұрыс емес. Батырма шын мәнінде мән орныққанша жоғары және төмен жүреді (оң жақтағы график). Енді, батырманың қалай әрекет ететінін біле отырып, батырманың өзгеру күйін тіркейтін, бірнеше уақыт күтетін, содан кейін ауыстырып-қосқыштың күйін қайта оқитын, серпілісі бар батырма үшін бағдарлама жазуға болады.
Мұндай бағдарламаның алгоритмін келесідей жазуға болады:
1. Батырманың алдыңғы және ағымдағы күйін сақтаймыз (LOW орнату кезінде).
2. Батырманың ағымдағы күйін оқимыз.

3. Батырманың ағымдағы күйі алдыңғысынан өзгеше болса, 5 мс күтеміз, себебі, түйме өз күйін өзгертуі мүмкін.
4. 5 мс күткеннен кейін батырманың күйін оқып, оны батырманың ағымдағы күйіне айналдырамыз.
5. Батырманың алдыңғы күйі LOW, ағымдағы күйі HIGH болса, жарықдиодының күйін ауыстырып қосамыз.
6. Батырманың алдыңғы күйін ағымдағы күй ретінде орнатыңыз.
7. 2-қадамға оралыңыз.

Бұл алгоритм функцияларды меңгерудің тамаша үлгісі болып табылады. Функция - кіріс аргументтерін қабылдай алатын, оларды қолданып код үзіндісін орындайтын және нәтижені қайтаруы мүмкін оператор. Мұны білмей-ақ, бағдарламалардағы функцияларды кездестірдіңіз. Мысалы, digitaiwrite () аргумент ретінде контакт нөмірін және мәнді (HIGH немесе LOW) қабылдайтын және сол мәнді контактіге орнататын функция.

Бағдарламаны орындау процесі қадамдарды көп рет қайталау болып табылады. Бірнеше рет шақыруға болатын, контактінің серпілісін жою функциясын жазайық. Функция батырманың алдыңғы күйін кіріс деректері ретінде қабылдайды, серпілуден қорғауды орындайды және батырманың орнатылған тұрақты күйін шығарады. Бағдарламаның негізгі циклі, батырманы әрбір басқан сайын жарықдиодының күйін ауыстырып қосады. 2 листинг кодын Arduino платасына жүктеп, оның қалай жұмыс істейтінін көріңіз.

Листинг 2. Батырманың серпілуін жою

```
const int LED=9;                    // 9 контакт жарықдиодын қосу үшін
const int BUTTON=2;                 // 2 контакт батырманы қосу үшін
boolean lastButton = LOW;           // Батырманың алдыңғы күйін
                                    // сақтауға арналған айнымалы
boolean currentButton = LOW;        // Батырманың ағымдық күйін
                                    // сақтауға арналған айнымалы
boolean ledOn = false;              // Жарықдиодының ағымдық күйі
                                    // қосулы / сөндірулі

void setup()
{
  pinMode (LED, OUTPUT);            // Жарықдиодының контактісін
                                    // шығыс ретінде орнату
  pinMode (BUTTON, INPUT);          // Жарықдиодының контактісін
                                    // кіріс ретінде орнату
}
/*
* серпілуді реттеу функциясы
* аргумент ретінде батырманың алдыңғы күйін қабылдайды
* және нақты мәнді шығарады
*/
boolean debounce(boolean last)
{
  boolean current = digitalRead(BUTTON);   // Батырманың күйін оқу
  if (last != current)                     // Егер өзгерсе...
```

```
  {
    delay(5);                              // 5 мс күтеміз
    current = digitalRead(BUTTON);         // Батырманың күйін оқиды
    return current;                        // Батырманың күйін қайтарады
  }
}

void loop()
{
  currentButton = debounce(lastButton);
  if (lastButton == LOW && currentButton == HIGH)  // Егер басу
  {
    ledOn = !ledOn;                                // Жарықдиоды күйінің
                                                   // мәнін теріcтеу
  }
  lastButton = currentButton;
  digitalWrite(LED, ledOn);                        // Жарықдиоды күйінің
                                                   // статусын өзгерту
}
```

Енді 2 листингтегі кодты толығырақ қарастырайық. Алдымен батырманы және жарықдиодты қосуға арналған контактілердің нөмірлері берілді. Содан кейін бағдарламада өзгеретін үш ауқымды логикалық айнымалылар жарияланады (ауқымды айнымалының мәнін бағдарламаның кез келген бөлігінде өзгертуге болады). Үш айнымалының әрқайсысына бастапқы мәндер (LOW, LOW және false) меншіктелген. Ары қарай бағдарламада бұл айнымалылардың мәндерін меншіктеу операторы = арқылы өзгертуге болады.

boolean debounce () батырмасының серпілуін жою функциясын қарастырайық. Бұл функция батырманың алдыңғы күйінің логикалық айнымалысын (тек екі күйі бар: true/false, HIGH/LOW, қосу/сөндіру, 1/0) қабылдайды және батырманың күйінің ағымдық мәнін қайтарады. Функцияның ішінде != (тең емес) операторының көмегімен батырманың ағымдық күйі алдыңғысымен салыстырылады. Күйлер әртүрлі болса, батырма басылған болуы мүмкін. Содан кейін батырманың күйін қайта тексермес бұрын, 5 мс күтеміз (бұл батырманың күйі серпілгеннен кейін тұрақтануы үшін жеткілікті). Сосын батырманың күйін қайта тексереміз. Естеріңізде болса, функциялар нәтижені қайтара алады. Бұл функция тек debunce () функциясында жарияланған және қолданылатын логикалық жергілікті айнымалының ағымдағы мәнін қайтарады. debounce () функциясы негізгі циклден шақырылған кезде, қайтарылған мән бағдарламаның басында анықталған currentButton ауқымды айнымалысына жазылады.

debounce () функциясын шақырғаннан және currentButton айнымалысының мәнін орнатқаннан кейін, && операторының («И» логикалық операторы, егер && операторы бөлген теңдіктердің әрқайсысы шын болса ғана, жақшадағы өрнек орындалатынын білдіреді) көмегімен батырманың ағымдағы және алдыңғы күйінің мәндері салыстырылады.
Егер батырманың күйі осыған дейін LOW, енді HIGH болса, бұл батырма басылғанын және ledOn айнымалысының мәнін өзгерту керек екенін білдіреді. Бұл әрекетті ledon айнымалысының алдында ! операторы орындайды. Цикл аяқталды, батырманың алдыңғы күйі айнымалысын жаңартамыз және жарықдиодының күйін өзгертеміз. Бағдарлама батырманы әр басқаннан кейін жарықдиодының күйін өзгертеді.

RGB жарықдиодында басқарылатын түнгі жарық жасау

Arduino платасына үш түсті RGB жарықдиодын қосып, батырманы басқан кезде түсі өзгеретін түнгі жарық жасаңыз. RGB жарық диодында олардың әрқайсысының жарықтығын, импульстік модуляциялау арқылы түстерді араластыруға болады.
Құрылғыда біз төрт шығысы бар RGB жарықдиодын қолданамыз, олардың біреуі барлық үш диодқа ортақ катод, ал қалғандары әр түсті диодтарға арналған анодтар. 3 суретте көрсетілгендей, RGB жарықдиодын Arduino платасының үш ЕИМ (PWM) контактілеріне ток шектейтін резисторлар арқылы қосыңыз.

Батырманы басқан сайын жарықдиодты түстердің циклдік ауысуын реттеуге болады. Бұл жағдайда жарықдиодты түстерді келесі күйге қою үшін функцияны қосу ыңғайлы. 3 листингте ұсынылған бағдарлама жеті түсті және жарықдиоды жанбаған күйді анықтайды. analogwrite () функциясын қолдана отырып, өз түстеріңіздің комбинацияларын орнатуға болады. loop () циклінің алдыңғы мысалдан жалғыз айырмашылығы - жарықдиоды күйлері санының артуы (шеңбер бойынша 0-ден 7-ге дейін). Бағдарламаны платаға жүктеп, түрлі-түсті түнгі жарықпен тәжірибе жасаңыз. Analogwrite () функциясының мәндерін өз мәндеріңізге өзгерту арқылы RGB жарықдиодының түсін өзгертіңіз.

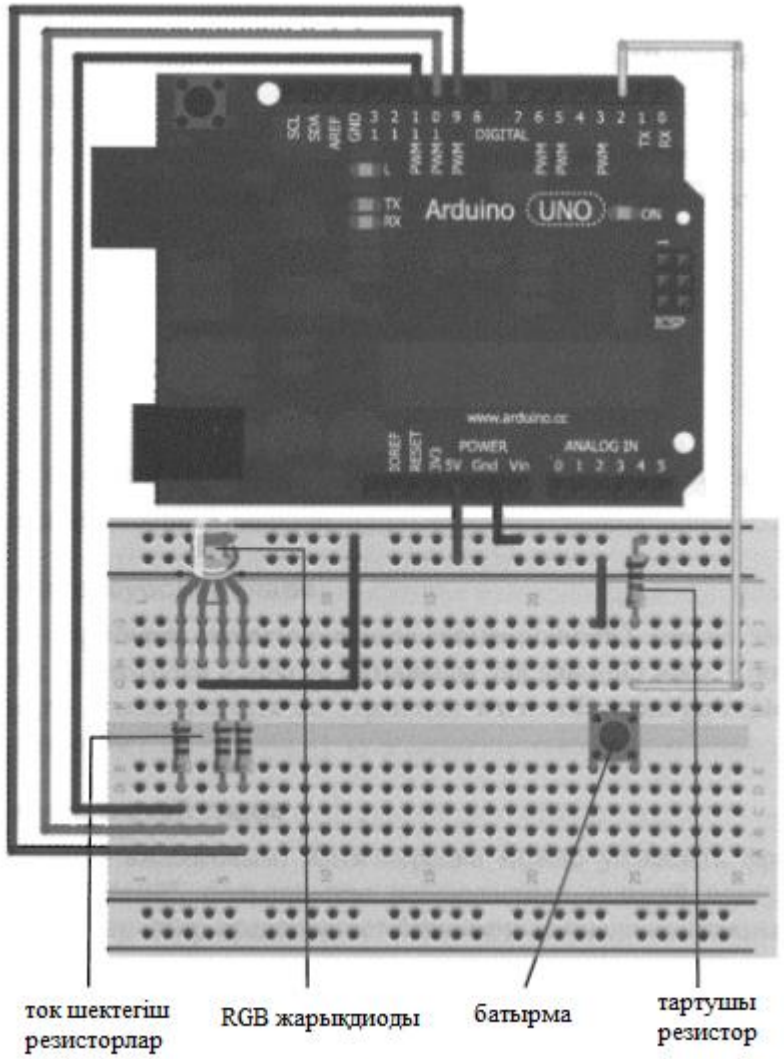
Листинг 3. Жарықдиодында басқарылатын түнгі жарық

```
const int BLED=9;           // 9 контакт BLUE RGB-жарықдиоды шығысы үшін
const int GLED=10;          // 10 контакт GREEN RGB - жарықдиоды шығысы үшін
const int RLED=11;          // 11 контакт RED RGB- жарықдиоды шығысы үшін
const int BUTTON=2;         // 2 контакт батырма кірісі үшін

boolean lastButton = LOW;     // Батырманың алдыңғы статусы
boolean currentButton = LOW; // Батырманың ағымдық статусы
int ledMode = 0;              // RGB-жарықдиоды статусының мәні

void setup()
{
  pinMode (BLED, OUTPUT);      // жарықдиодының BLUE контактісін
                               // шығыс ретінде орнату



  pinMode (GLED, OUTPUT);      // жарықдиодының GREEN контактісін
                               // шығыс ретінде орнату
  pinMode (RLED, OUTPUT);      // жарықдиодының RED контактісін
                               // шығыс ретінде орнату
  pinMode (BUTTON, INPUT);     // Батырманың контактісін
                               // кіріс ретінде орнату
```

```
/*
*  серпілуді реттеу функциясы
*  аргумент ретінде батырманың алдыңғы күйін
*  қабылдайды және нақты мәнін шығарады
*/
boolean debounce(boolean last)
{
  boolean current = digitalRead(BUTTON);   //  Батырма күйін санау
  if (last != current)                     //  Егер өзгерсе ...
  {
    delay(5);                               //  5 мс күтеміз
    current = digitalRead(BUTTON);          //  Батырманың күйін оқимыз
    return current;                         //  Батырманың күйін
                                            //  қайтарамыз
  }
}

/*
* жарықдиоды режимін таңдау
* режим нөмірін жіберу және жарықдиодының берілген режимін орнату
*/
void setMode(int mode)
{
  //  қызыл
  if (mode == 1)
  {
    digitalWrite(RLED, HIGH);
    digitalWrite(GLED, LOW);
    digitalWrite(BLED, LOW);
  }
  //  жасыл
  else if (mode == 2)
  {
    digitalWrite(RLED, LOW);
    digitalWrite(GLED, HIGH);
    digitalWrite(BLED, LOW);
  }
  //  көк
  else if (mode == 3)
  {
    digitalWrite(RLED, LOW);
```

```
    digitalWrite(GLED, LOW);
    digitalWrite(BLED, HIGH);
  }
  //  күлгін (қызыл + көк)
  else if (mode == 4)
  {
    analogWrite(RLED, 127);
    analogWrite(GLED, 0);
    analogWrite(BLED, 127);
  }
  // көгілдір (көк + жасыл)
  else if (mode == 5)
  {
    analogWrite(RLED, 0);
    analogWrite(GLED, 127);
    analogWrite(BLED, 127);
  }
  // қызғылт сары (жасыл + қызыл)
  else if (mode == 6)
  {
    analogWrite(RLED, 127);
    analogWrite(GLED, 127);
    analogWrite(BLED, 0);
  }
  // Ақ (жасыл + қызыл + көк)
  else if (mode == 7)
  {
    analogWrite(RLED, 85);
    analogWrite(GLED, 85);
    analogWrite(BLED, 85);
  }
  //  сөндірулі (mode = 0)
  else
  {
    digitalWrite(RLED, LOW);
    digitalWrite(GLED, LOW);
    digitalWrite(BLED, LOW);
  }
}

void loop()
{
  currentButton = debounce(lastButton);           //  батырма статусын
                                                  //  оқу
  if (lastButton == LOW && currentButton == HIGH) //  егер батырма басулы болса
  {
```

```
    ledMode++;                                  //  жарықдиоды статусының
                                                //  айнымалы өсімі


  }
  lastButton = currentButton;
                                        //  Цикл бойынша жарықдиодының
                                        //  жарқырауының барлық режимдері өтті
                                        //  бастапқы нұсқаға түсіру =0
  if (ledMode == 8)
    ledMode = 0;
  setMode(ledMode);                      //  жарықдиоды режимін өзгерту
}
```



сурет 3. Түнгі жарықтың қосу схемасы