Робота з аналоговими портами

На відміну від цифрового, аналоговий вхід Arduino приймає сигнали від 0 до 5В. Але для того, щоб з цим сигналом можна було працювати далі, він оцифровується. Для цього служить АЦП – аналого-цифровий перетворювач з розрядністю 10 біт. Це означає, що коли на вході АЦП напруга змінюється від 0 до 5В, на виході АЦП маємо значення від 0 до 1023. Тобто $2^{10} = 1024$ різних значення. А щоб получити значення напруги, потрібно порахувати просту пропорцію:

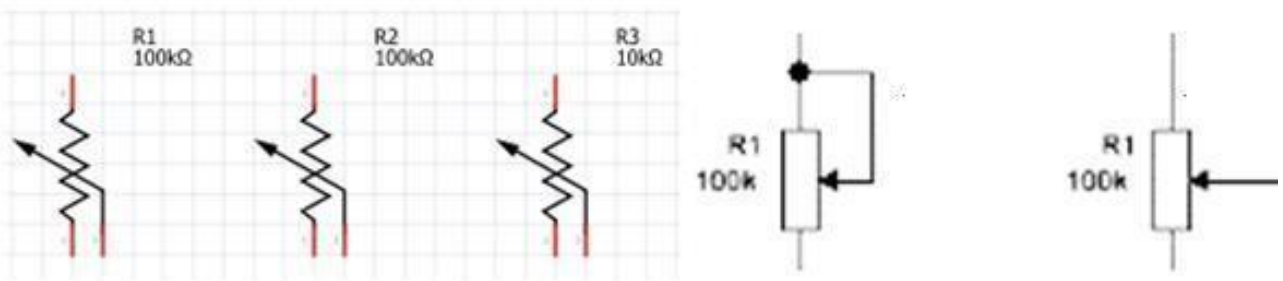
$$V_{real} = \frac{значення\ з\ АЦП * 5}{1024}$$

Значення 5В може трохи змінюватися, в залежності від блоку живлення Arduino.

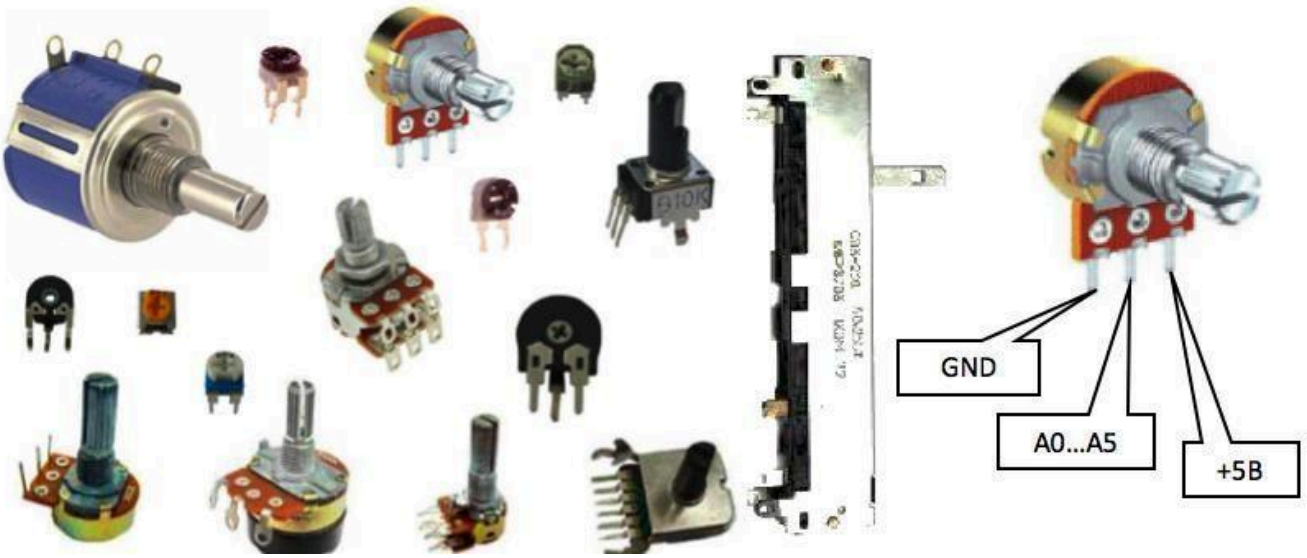
Аналогові порти використовують для прийняття сигналів з аналогових датчиків, таких як датчики звуку, світла, джойстики, а також змінні резистори або потенціометри.

Роздивимося останні. На малюнках виглядають вони ось так:

На принциповій схемі позначаються так:



Реальні виглядають ось так:



Два крайні виводи підключаємо до землі GND та піну +5В. Середній вивід можна підключати до будь-якого аналогового порту A0…A5.

Ознайомимося з основними функціями для опрацьовування аналогового сигналу.

Читання аналогового сигналу:

Функція **analogRead(pin);** зчитує аналоговий сигнал (оцифровує), де

pin – номер Аналогового піна: 0-5 (Для Arduino UNO).

Функція повертає значення 0.. 1023, в залежності від напруги на піні від 0 до приблизно 5 В.

Змінення діапазону значень:

Функція **map(val, min, max, new_min, new_max);** повертає величину в новому діапазоні, де

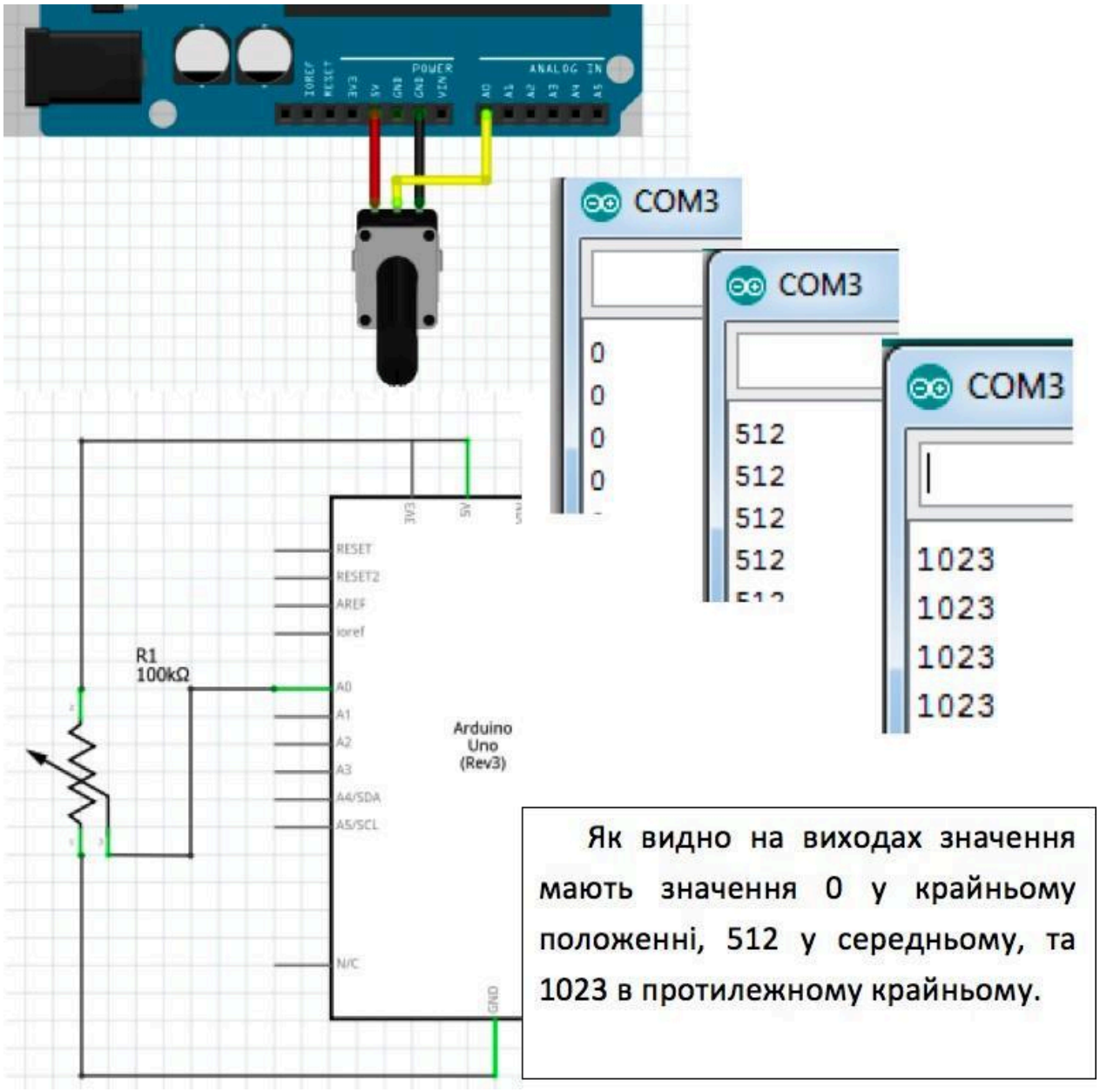
val – вхідна величина.

min, max – мінімальне і максимальне значення при вході в **map.**

new_min, new_max – відповідно, мінімальне і максимальне значення на виході.

Функція **constrain(val, min, max);** обмежує діапазон змінної **val** до вказаних значень **min** і **max.**

Побудуємо схему:



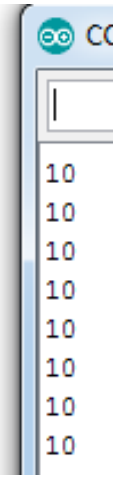
Напишемо скетч:

```
#define potent_pin 0 //Сюди підключена середня ніжка потенціометра
int val;             //Змінна для зберігання значень потенціометра
void setup() {
  Serial.begin(9600);
}

void loop() {
  val = analogRead(potent_pin);//Запам'ятати значення з потенціометра
  Serial.println(val);//Вивести в порт
  delay(30);
}
```

Зменшимо діапазон цифрових значень з 0 до 10, використавши вивчені функції:

```
#define potent_pin 0   // Сюди підключена середня ніжка потенціометра
int val;               // змінна для зберігання значень потенціометра
void setup() {
  Serial.begin(9600);
}
void loop() {
  val = analogRead(potent_pin);  // запам'ятати значення з потенціометра
  val = map(val, 0, 1023, 0, 10);// перевести в діапазон 0.. 10
  val = constrain(val, 0, 10); // обмежити діапазон 0.. 10
  Serial.println(val);              // вывести в порт
  delay(30);
}
```

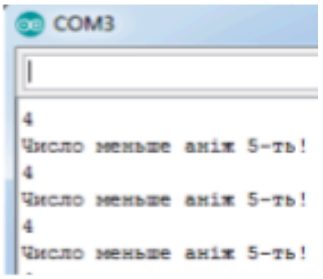


Функції **map** і **constrain** звичайно використовують в парі. Тепер ми легко зможемо регулювати кут повертання сервоприводу, яскравість світла, швидкість обертання, потужність нагріву, силу утримання електромагніту, і навіть кількість світлодіодів, що світяться!

Якщо додати просту умову, то можна дізнатися про перевищення якогось порогового значення з датчика.

У наступному прикладі навчимо Arduino повідомляти нам, чи число більше, чи менше аніж п’ять з діапазону 0…10 виводиться у COM порт:

```
#define potent_pin 0 //сюди підключена середня ніжка потенціометра
int val;             //змінна для зберігання значень потенціометра
void setup() {
  Serial.begin(9600);
}
void loop() {
  val = analogRead(potent_pin);//запам'ятати значення з потенціометра
  val = map(val, 0, 1023, 0, 10);    //перевести в діапазон 0...10
  val = constrain(val, 0, 10);          //обмежити діапазон 0...10
  if(val>5)Serial.println("Число більше 5-ти!");  //вивести в порт
  else Serial.println("Число меньше аніж 5-ть!"); //вивести в порт
  Serial.println(val);                           //вивести в порт
  delay(30);
}
```

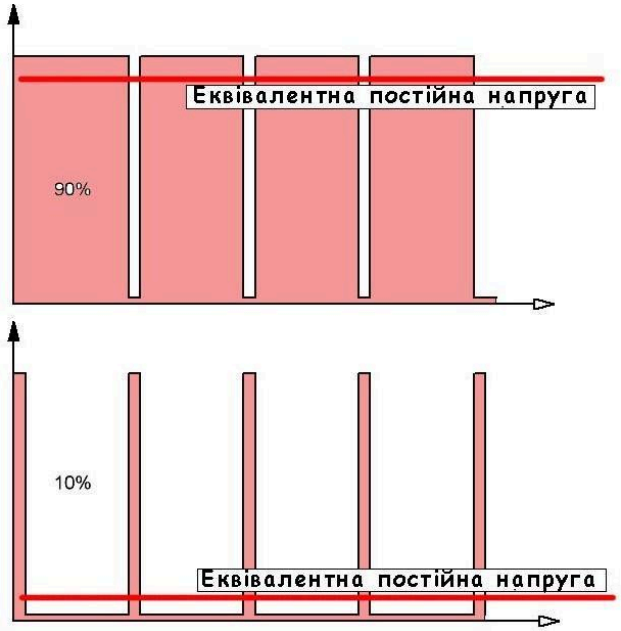


Таким чином, можна, наприклад, впіймати момент про перевищення рівня з датчика звуку або світлового потоку з датчика світла та вирішити інші задачі з використанням порогових значень.

Широтно-Імпульсна Модуляція (ШІМ, PWM)

Після вивчення попередніх тем перед нами обов’язково постане питання, як можна регулювати потужність споживача, наприклад, керувати яскравістю світлодіода або регулювати швидкість двигуна. Найпростіший спосіб – послідовно навантаженню, наприклад, – світлодіоду, включити резистор. Але ж він буде грітися і забирати дорогоцінну енергію, і чим потужніше світлодіод, тим сильніше буде грітися резистор. Такий варіант нам не підходить. А давайте спробуємо дуже швидко вмикати і вимикати світлодіод, при цьому змінюючи тривалість включень! Наприклад, якщо включати світлодіод на 0,5 мілісекунд кожну мілісекунду, то світлодіод засвітиться, але не на повну яскравість. Аналогічно з двигуном – включати двигун на 30 секунд кожну хвилину – тоді двигун розкрутиться, але не на повну швидкість, останні 30 секунд він буде рухатися по інерції, та гальмуватися за рахунок сили тертя. Таким чином, двигун буде крутитися на 50 відсотків своєї потужності.

Широтно-імпульсна модуляція (ШІМ, PWM)



ШІМ є імпульсний сигнал постійної частоти і змінної скважності, тобто відношення періоду проходження імпульсу до його тривалості. З допомогою завдання скважності (тривалості імпульсів) можна міняти середню напругу на виході **ШІМ**. У цифровій техніці, виходи якої можуть приймати тільки одне з двох значень, наближення бажаного середнього

рівня виходу за допомогою ШІМ є абсолютно природним. Давайте на практиці спробуємо змінювати яскравість світлодіода. Але спочатку ознайомимося з функціями та особливостями Arduino, при роботі з ШІМ сигналами:

Генерація ШІМ-сигналу відбувається за допомогою функції:

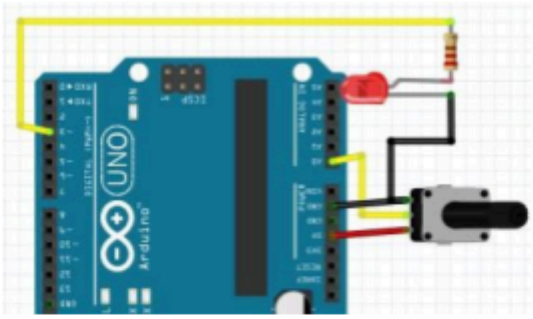
analogWrite(pin, duty);

pin – пін, на якому генерується ШІМ

duty – величина 0.. 255, відповідає значенню ШІМ 0.. 100%.

ШІМ-піни Arduino NANO, UNO: 3, 5, 6, 9, 10, 11 (інші моделі – дивись інструкцію).

А тепер змонтуємо макет з використанням світлодіода, резистора, що обмежує струм, та потенціометра, за допомогою якого будемо змінювати скважність сигналу на на ШІМ піні 3 Arduino:



Напишемо скетч:

```
int pwm;
void setup() {
}
void loop() {
 pwm = analogRead(0);
 pwm = map(pwm, 0, 1023, 0, 255);
 pwm = constrain(pwm,3,pwm);
 analogWrite(3, pwm);
}
```

Змінній цілого типу **pwm** присвоюється значення з аналогового входу A0, яке змінюється потенціометром від 0 до 1023. Функція **map** змінює цей діапазон до

значень 0…255. Функція constrain на всяк випадок обрізає цей діапазон. Після цього змінна **pwm** передається на ШІМ вихід на 3 пін. Якщо тепер покрутити потенціометр, світлодіод буде змінювати яскравість світла!