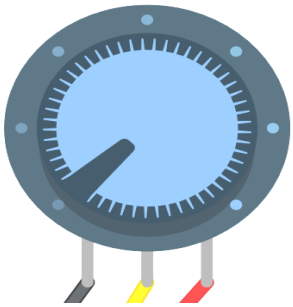
Arduino koolituse kolmas tund – keerame nuppu

Õppematerjali koostajad: Indrek Karo ja Angela Leppik

Potentsiomeeter

Potentsiomeeter on muuttakisti. Põhimõtteliselt on tegemist kahe erineva takistusega takistiga, mille vahel jagatakse pinget. Peamiselt on see kasutuses elektroonikas signaali sujuvaks reguleerimiseks (näiteks helitugevuse reguleerimiseks), erinevalt reostaadist, mis on kasutusel tugevvoolutehnikates voolutugevuse muutmiseks (näiteks elektrimootorite pöörlemiskiiruse reguleerimiseks). Arduino õppimisel on potentsiomeetrit hea kasutada analoogsignaali kasutamise õppimisel. Potentsiomeetril on kolm ühenduspunkti (kaks takisti oma ning üks siis signaali reguleerimiseks). Seda on ka näha pildil nr 24. Kus punane juhe on vastavalt 5V sisend takistitele, must on maanduse ning kollane signaali jaoks.



Pilt nr. 24

Potentsiomeetrit saab kasutada Arduino puhul analoogsignaali töötlemise harjutamiseks. Analoogsignaal on signaal, millel on lõputu arv olekuid, mida saab mõõta. Graafiliselt võiks seda näiteks kirjeldada sinusoidina, nagu on pildil nr. 25, kus x muutub ajas „(t)“. Reaalses maailmas võib näitena tuua räägitava kõne, kus nii helitugevus, kui ka sagedus muutuvad.

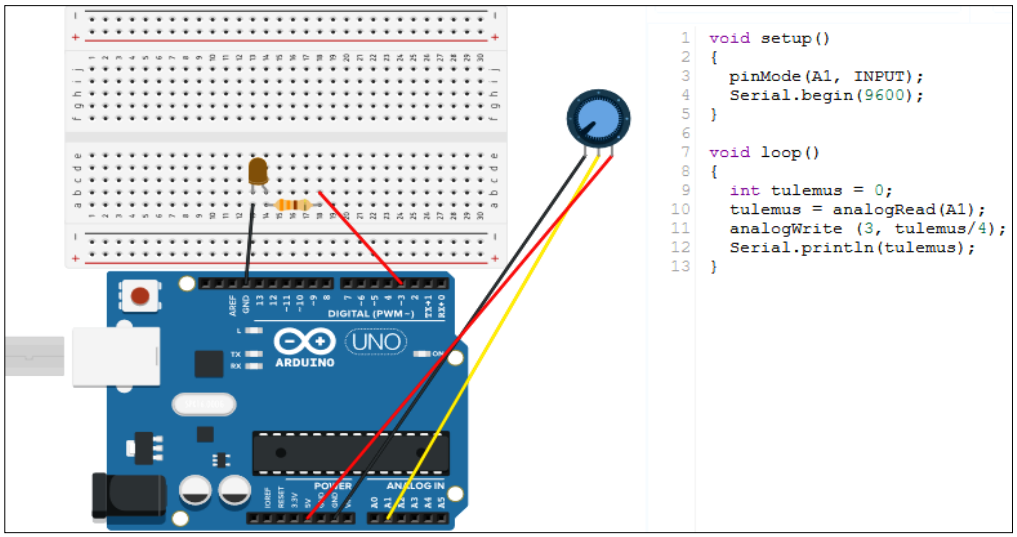
Pilt nr. 25

Kuna Arduino tegutseb digitaalses maailmas, kus väärtused on kas „1“ või „0“, siis peab Arduinole selgeks tegema analoogsignaali vastuvõtmise. Kuna Arduino analoogsisend on 10-bitine, siis ta omab 1024 väärtust (0-1023), seega saab 5V pinge jagada 1024 osaks. Näiteks, kui pinge on 2,5 volti, siis väärtus on 512 ning kui 1,25 volti siis on väärtuseks 256 jne. See ei ole teatavate füüsikaliste iseärasuste ning elektroonika kasutusloogika järgi 100% täpne, kuid sobib hästi tööpõhimõtte näitamiseks ning lihtsamate skeemide tegemiseks.

Analoogsignaali lugemiseks on kasutusel käsk „analogRead ()“, kus sulgudes on ära märgitud pesa, kuskohast seda signaali loetakse. Tavaliselt on selleks pesad A0–A5. Need pesad on võimelised vastu võtma 10-bitist (2^{10}–väärtust) signaali.

Kuna Arduino töötab siiski digitaalse süsteemina, siis on vaja välja saadetava signaali puhul tekitada illusioon analoogsignaalist. Selleks kasutatakse pulsilaiusmodulatsiooni käsuga „analogWrite ()“, kus sulgudes on pesa, millest signaali välja saadetakse. Selleks sobivad ainult need pesad, millel on ees „kapsaraua“ märk („~“), neid nimetatakse ka „PWM“ väljaviikudeks. Tegelikult on digitaalses maailmas ikkagi väärtus kas „1“ või „0“. Kuna lülitused on väga kiired (signaal pulseerib, sellest ka nimi) ja viited on sobivaks seadistatud, siis inimsilm näeb asja erineva intensiivsusena ning selleks ongi kasutusel pulsilaiusmodulatsioon.

„PWM“ väljaviikude häda on see, et need on võimelised välja saatma ainult 8-bitist (256 ehk $2^{8)}$ väärtust, mistõttu on meil vaja teha väike teisendus ning jagada vastu võetud signaal arvuga 4, saamaks soovitud tulemust.



Pilt nr. 26

Potentsiomeetri toimimise katsetamiseks loodud skeem koos programmikoodiga asub pildil nr. 26.

Pildil nr. 26 on toodud skeem mille abil saab reguleerida potentsiomeetri abil valgusdioodi heledust. Selleks on ühendatud Arduino väljaviiguga nr. 3 takisti (330 oomine) ja LED-lambike. Samuti on ühendatud skeemiga potentsiomeeter. Viimase puhul on mõistlik tähele panna, et keskmine kontakt on kasutusel signaali jaoks ning äärmised siis 5V ja maandus. Potentsiomeetri puhul muutub signaal selle peal olevat „seierit“ keerates. Kui seier on maanduse pool, siis on signaal nõrgem ning 5V pool, siis on tugevam. Kui 5V ja maanduse ühendused potentsiomeetril ära vahetada siis peab lihtsalt keerama seierit teisele poole ning tulemus on sama.

Koodi vaadates on näha, et on kasutatud jadapordi monitori. Põhjus on selles, et algselt ei ole teada kuhu poole tuleb potentsiomeetrit keerata, et tulemus suureneks ning siis saame vaadata, kuidas potentsiomeetrist tulev signaal muutub.

Pilt nr. 26 koodi selgitus

```
void setup ()
{
 pinMode (A1, INPUT);        //hakkab kasutama väljaviiku A1, saamaks signaali
 Serial.begin (9600);        //hakkab vahetama läbi jadapordi andmeid sulgudes märgitud kiirusel
}

void loop ()

{
```

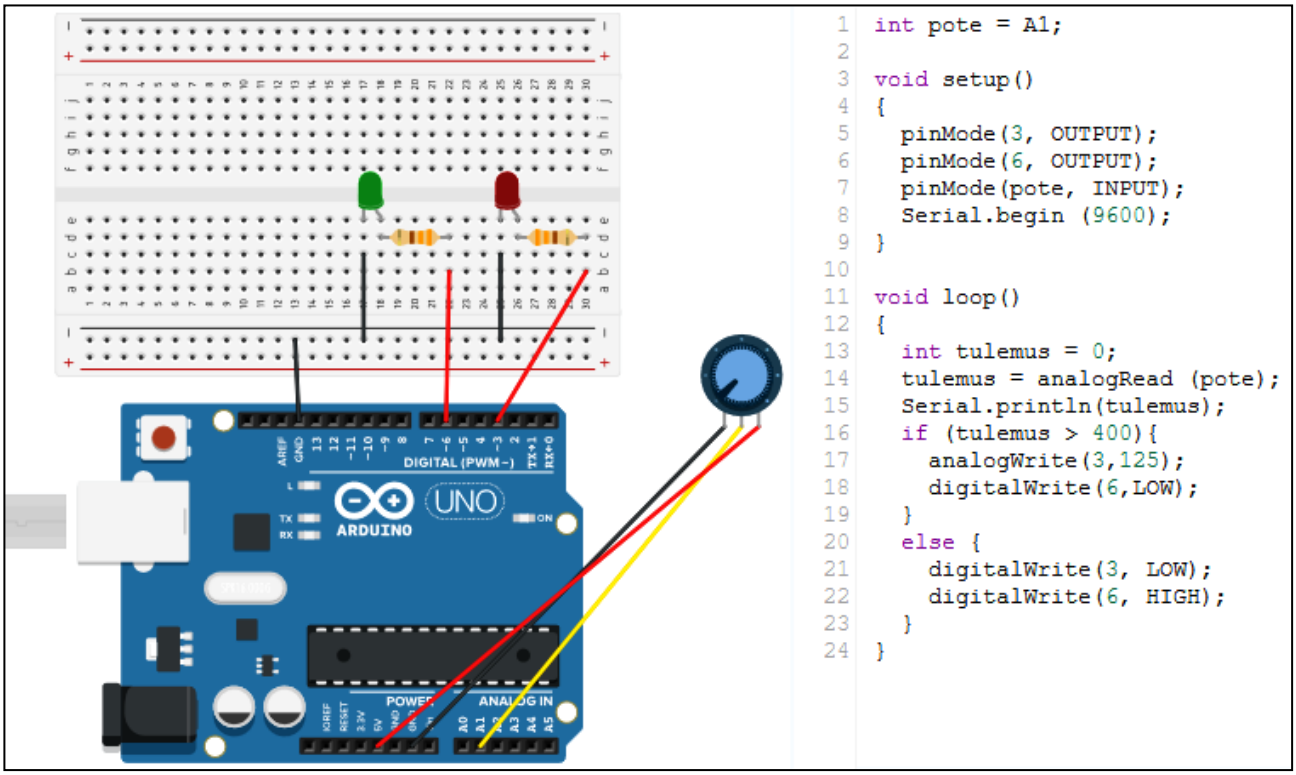
int tulemus = 0; //luuakse lokaalne muutuja „tulemus“ mille väärtus on algselt „0“, see on vajalik, kuna saadud väärtust on vaja jagada neljaga, sest pesa 3 ei ole võimeline välja saatma rohkem kui 8-bitist (256 väärtust) signaali, aga sisse tulev signaal on 10-bitine (1024 väärtust) ning muutuja jagamine mingi arvuga teeb koodi lihtsamalt loetavaks (see ei ole hädavajalik, saab ka ilma).

tulemus = analogRead (A1); //siin antakse teada millised väärtused hakkavad olema muutujal „tulemus“, sellepärast on ka algselt väärtus „0“.

analogWrite (3, tulemus/4); //siin kasutatakse pulsilaiusmodulatsiooni ehk tekitatakse illusioon analoogsignaalist ning see omakorda muudab pesasse 3 ühendatud LED heledust (analogWrite). Kuna aga antud pesa ei ole võimeline välja saatma rohkem kui 256 erinevat väärtust, siis on vaja muutuja „tulemus“ jagada arvuga 4 (1024/4=256) ning siis saame sobiva arvu väärtusi (väärtus 256).

Serial.println (tulemus); //see rida võimaldab jadapordi monitori abil jooksvalt vaadelda muutuja „tulemus“ erinevaid väärtusi niiviisi, et iga väärtus oleks omaette real ning on näha, kuidas vastavalt tulemustele muutub valgusdioodi heledus.

Potentsiomeetrit saab kasutada ka mitme erineva valgusdioodi juhtimiseks. Selle jaoks on mitmeid erinevaid käske, kuid hetkel käsitleme neist vaid lihtsaimat võimalust – „if“ lauset. Põhimõtteliselt saab ju teha näiteks kahe LED lambiga skeemi, kus üks lamp hakkab põlema siis, kui potentsiomeetri väärtus on läinud liiga suureks. Üks selline skeem koos koodiga on ära toodud pildil nr. 27.



Pilt nr. 27

Antud skeemi puhul on kasutusel globaalne muutuja „pote“, tähistamaks pesa, kuhu on ühendatud potentsiomeetri signaali juhe. See ei ole hädavajalik, aga tihtipeale keerulisematel skeemidel võib kippuda ununema, kuhu on ühendatud näiteks potentsiomeeter (füüsilisel kujul eriti, sest juhtmeid võib olla palju ning neid ei leia alati üles). Tinkercadi simulatsioonil on asi natuke lihtsam.

```
int pote = A1;                          //deklareeritakse muutuja „pote“, tähistamaks potentsiomeetri signaalijuhtme väljaviiku

void setup ()
{
pinMode (3, OUTPUT);                    //antakse Arduinole teada, et pesast 3 saadetakse voolu välja
pinMode (6, OUTPUT);                    //antakse Arduinole teada, et pesast 6 saadetakse voolu välja
pinMode (pote, INPUT);                  //antakse Arduinole teada, et loetakse sisendit pesast nimega „pote“
Serial.begin (9600);                    //pannakse tööle jadapordimonitor sulgudes märgitud kiirusega
}
void loop ()
{
int tulemus = 0;                        //siin tutvustatakse Arduinole muutujat „tulemus“ algväärtusega 0
tulemus = analogRead (pote);            //siin näidatakse ära, et „tulemus“ hakkab muutuma vastavalt potentsiomeetrist tulevale signaalile
Serial.println (tulemus);               //see kirjutab jadapordi monitoris välja potentsiomeetrist loetava tulemuse, et saaksime vaadata millal lamp süttib
if (tulemus > 400) {                    //„if“ lause algus, kui tulemus on üle 400, siis hakkab toimuma järgnev tegevus
analogWrite (3, 125);                   //hakkab põlema punane lamp poole võimsusega. Siin on kasutatud käsku „analogWrite“ näitamaks, et saab ka lihtsalt määrata, kui heledalt lamp põleb.
digitalWrite (6, LOW);                  //kustub roheline LED.
}                                       //„if“ lause lõpp.
else {                                  //tegevus, mis toimub, siis kui ei ole täidetud „if“ lause tingimus
digitalWrite (3, LOW);                  //punane lamp ei põle
digitalWrite (6, HIGH);                 //põleb roheline lamp
}                                       //selle tegevuse lõpp, mis toimub siis, kui „if“ lause tingimus ei ole täidetud

}                                       //korduse lõpp, peale seda algab kordus otsast peale
```

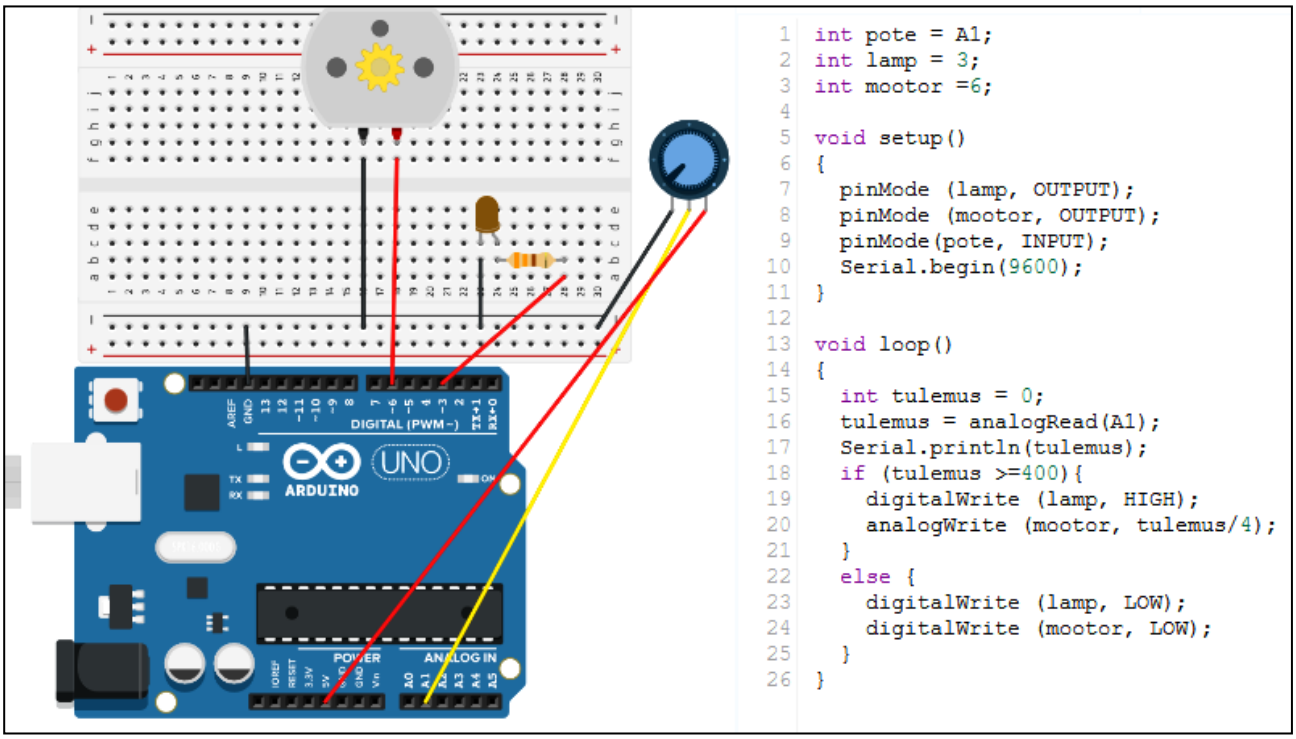
Sellist kahe lambiga süsteemi võib vaadata kui näidet hoiatuslampidest. Kui signaal ületab mingi piiri, siis hakkab tööle punane hoiatuslamp, muudel juhtudel põleb roheline tuli. Sama süsteemi saab rakendada ka rohkemate lampidega, lihtsalt analoogsignaali eri väärtustel süttivad, kas erinevad lambid või näiteks rohkem lampe. Viimase näitena võib vaadata helitugevuse reguleerimise potentsiomeetrit (mida valjemaks heli läheb, seda rohkem lampe põleb).

Sarnast vastuvõetava analoogsignaali digitaalsel kuju töötlemist saab kasutada erinevate tegevuste reguleerimiseks. Näiteks saab tunnis teha mootori sisse lülitamise ja kiiruse reguleerimise potentsiomeetriga, mis on ära toodud pildil nr. 28. Antud skeem on näiteks, et mootori kiirust saab reguleerida enam-vähem iga analoogsisendiga. Näiteks võib teha masina, kus ventilaatori kiirus sõltub temperatuurist. Mida kõrgem on temperatuur seda kiiremini pöörleb ventilaator ning tööle hakkab asi alles siis, kui on piisavalt palav. Antud skeemil on ka valgusdiood, mis näitab ära, et mootor peaks olema tööle hakanud. Koodi poolest meenutab pildil nr. 28 toodud süsteem eelmist kahe lambiga seadet, sisaldades nii globaalseid kui ka lokaalseid muutujaid, kuid teise lambi asemel on kasutusel hoopis muutuva kiirusega mootor.

Pildil nr. 28 koodi selgitus

```
int pote = A1;          //potentsiomeeter on ühendatud pesasse A1
int lamp = 3;           //valgusdioodi pesa on 3 ning kannab nime lamp
int mootor = 6;         //mootori tööle paneku pesa on 6 ning kannab nime mootor
```

Antud programmis ei pea kasutama globaalseid muutujaid, kuid see võib olla mõnel juhul mugavam, sest ei pea meeles pidama, millisesse väljaviiku erinevad asjad ühendatud on. Samuti on sellest kasu siis, kui on vaja tarvitada muutujaks määratud sisendit ja väljundit mitmes erinevas kohas.




```
int pote = A1;
int lamp = 3;
int mootor =6;

void setup()
{
  pinMode (lamp, OUTPUT);
  pinMode (mootor, OUTPUT);
  pinMode(pote, INPUT);
  Serial.begin(9600);
}

void loop()
{
  int tulemus = 0;
  tulemus = analogRead(A1);
  Serial.println(tulemus);
  if (tulemus >=400){
    digitalWrite (lamp, HIGH);
    analogWrite (mootor, tulemus/4);
  }
  else {
    digitalWrite (lamp, LOW);
    digitalWrite (mootor, LOW);
  }
}
```

Pilt nr. 28

Pildil nr. 28 oleva koodi funktsioonide selgitus

void setup () //antakse Arduinole teada, mida varem mainitud muutujatega tehakse
{
pinMode (lamp, OUTPUT);
pinMode (mootor, OUTPUT);
pinMode (pote, INPUT);
Serial.begin (9600); //käivitatakse jadapordi monitor sulgudes märgitud kiirusega
}

void loop () //algab programmi korduse osa
{
int tulemus = 0; //tekitatakse lokaalne muutuja „tulemus“, mis alguses on 0

tulemus = analogRead (A1); //tulemus hakkab muutuma vastavalt pesast A1 saadud signaalile (siia oleks võinud panna ka muutuja „pote“, kuidas kellelgi mugavam on)

Serial.println (tulemus); //näidatakse jadapordi monitoris potentsiomeetrist loetud tulemust

if (tulemus >= 400) {	//„if“ lause, kui tingimus on täidetud (muutuja „tulemus“ väärtus on suurem või võrdne 400-ga), siis hakkab toimuma „{„ja„}“ märkide vaheline tegevus
digitalWrite (lamp, HIGH);	//süttib LED
analogWrite (mootor, tulemus/4);	//hakkab tööle mootor, mille kiirus sõltub potentsiomeetrist saadavast tulemusest ning see jagatakse arvuga 4, sest väljundil saab olla ainult 256 väärtust (sest väljund on 8-bitine), aga sisend on ju 10-bitine (1024 väärtust)
}	//lõpeb see tegevus mis toimub siis kui muutuja nimega „tulemus“ on suurem või võrdne 400
else {	//siin on kirjeldatud tegevust, mis toimub siis, kui ei ole täidetud „if“ lause tingimused
digitalWrite (lamp, LOW);	//lamp ei põle, kui enne põles, siis kustub
digitalWrite (mootor, LOW);	//mootor ei pöörle
	//lõpeb ära see tegevus, mis toimub siis, kui „if“ lause tingimus ei ole täidetud
}	//lõpeb ära korduses toodud tegevus, et seejärel uuesti alustada

Seda ülesannet saab lahendada mitmetel eri meetoditel ning erinevate käskudega. Näiteks käsuga „MAP“ aga see on muu õppepäeva teema. Samuti ei pea antud ülesandes kindlasti kasutama muutujaid, ilma nendeta saab ka hakkama.