Arduino koolituse neljas tund – nimede andmine, muutujad ja mitmevärviline valgusdiood

Õppematerjali koostajad: Indrek Karo ja Angela Leppik

#define – käsuga antakse erinevatele programmis kasutatavatele konstantsetele väärtustele nimi. Eeliseks on see, et antud käsk ei võta ära mälu nagu näiteks muutujad, vaid kompilaator paneb selle programmi sisse sel hetkel kui on vaja. Probleem tekib defineerimisega tavaliselt siis, kui väärtusele pandav nimi on kuskil mujal programmi sees kasutusel. Näiteks mingi muu nimetuse osana, siis see osa tekstist või numbrist asendatakse #define käsuga määratuga, mis omakorda ajab kogu ülejäänud programmi segadusse.

Näide

#define p_pin 6

Siin asendatakse p_pin 6-ga kuna Arduino arvab, kui programmis on kirjas p_pin, siis selle all peetakse silmas pesa nr. 6.

#define käsul on mõningad olulised omadused, mida tuleb tähale panna ning mille poolest see käsk erineb muutujate deklareerimisest:

a) käsu lõpus ei tohi olla semikoolonit (märki „ ; “, sest siis hakkab Arduino arvama, et defineeritud nimi viitab näiteks mitte pesale 6, vaid hoopis pesal 6; ning siis üritab ta seda programmis üles leida;

b) ei tohi sisaldada „=“ märki, sest siis tekib eelnevalt kirjeldatuga samalaadne olukord.

Muutujad ja funktsioonid

Muutuja on oma olemuselt koht, kuhu andmeid salvestatakse. Sellel on nimi, väärtus ja tüüp. Muutujad jagunevad omakorda globaalseteks ja lokaalseteks muutujateks nagu on varasemas õppematerjalis kirjas. Globaalset muutujat on kaval kasutada siis, kui seda on vaja kogu programmi jooksul mitu korda, sest siis ei pea eraldi järgi vaatama, millele viidatakse. Lokaalset muutujat kasutatakse siis, kui seda on vaja mitu korda ühe kindla funktsiooni sees.

Koolituse läbiviimist toetas Eesti Teadushuvihariduse Liit ja Haridus- ja Teadusministeerium

Muutujaid saab eristada ka tüüpide järgi ning neid on mitmeid erinevaid. Järgnevalt on ära toodud mõned levinumad. Muid muutujatüüpe kirjeldatakse hiljem õppematerjalis jooksvalt.

Muutujad tuleb deklareerida (ehk Arduinole teatada muutuja tekitamisest) siis, kui neid kasutama hakatakse. Globaalsed muutujad deklareeritakse kohe programmi alguses enne funktsioone ning lokaalsed muutujad deklareeritakse funktsiooni sees enne nende kasutamist.

Muutuja deklareerimise näide

int lamp = 3;
Siin on muutujal ära määratud muutuja tüüp („int“ – tüüpi muutuja), nimi „lamp“ ja väärtus „3“, viimane tähendab Arduino puhul väljaviiku nr 3. Väärtusena kasutatakse sealt saadavat või sinna saadetavat signaali.

Muutujatele saab omistada väärtusi, mis tähendab seda, et muutuja väärtus ei ole kogu aeg sama.

Näiteks

lamp = digitalRead (3); siin omistatakse muutujale „lamp“ väärtus, mida ta loeb pesast 3.

Muutujatel on mitmeid tüüpe ning need erinevad tavaliselt suuruse ehk kasutatava mälumahu järgi. Seetõttu on ka osad muutujad rohkem kasutatavad. Peamiselt on kasutusel järgnevad muutujate tüübid:

int – tüüpi muutuja. See jätab meelde tavalise Arduino puhul kuni 16-bitise väärtuse (ehk väärtuse vahemikus -32,768 kuni 32,767) ning on üks kasutatavamaid muutujatüüpe. Sellega salvestatakse numbreid, nii positiivseid kui ka negatiivseid;

char – tüüpi muutuja. Seda kasutatakse ühe tähemärgi meelde jätmiseks muutuja kujul ning suuruseks on siis vähemalt 8 baiti. Need salvestatakse siiski ASCII tabeli järgi numbritena Arduino mällu ja seega saab nendega teha ka aritmeetilisi tehteid;

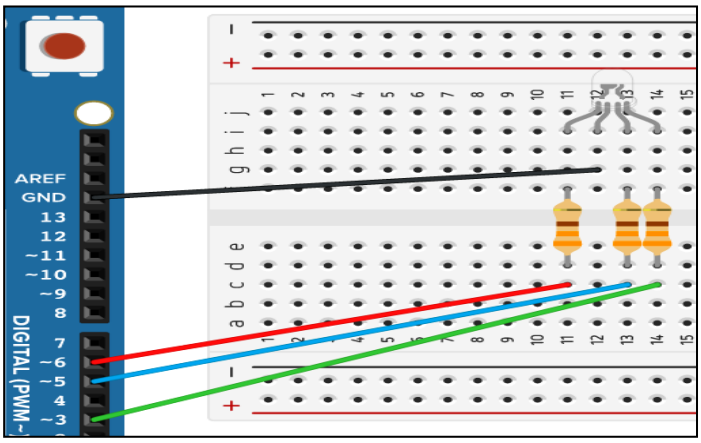
string – see on põhimõtteliselt teksti kujuline muutuja, tavaliselt on tegemist mingi sõnaga. Sellele viidatakse programmi koodis jutumärkidega (käesoleva õppematerjali viimases programmikoodis on ka näide). Kui on tegemist ühe tähemärgiga, siis on mõistlikum kasutada „char“ tüüpi muutujat.

Seda ei loeta tihtipeale ka muutujaks vaid hoopis tekstipõhiseks konstandiks, aga samas kasutatakse seda tihti koos muutujatega;

byte – jätab meelde numbri vahemikus 0-255, on muutujana kuni 8 biti suurune.

Mitmevärvilne LED

Katsetamaks käsku #define ning lokaalseid muutujaid, koostame skeemi kasutades mitmevärvilist valgusdioodi nagu on ära toodud pildil nr. 29.



Pilt nr. 29

Mitmevärvilise LED lambi omapära on see, et tegemist on põhimõttelistelt ühes kestas oleva 3 erineva LED lambiga, mida saab pulsilaiusmodulatsiooni abil erineva intensiivsusega põlema panna ning tekitada erinevat värvi valgust. Seetõttu on ka pildil nr. 29 oleval lambil kolm jalga, mida ühendatakse läbi takisti PWM väljaviikudesse ning üks jalg (katood, füüsilise dioodi puhul on see jalg teistest pikem) maandusesse. Antud pildil on erinevat värvi juhtmed, näitamaks millisest pesast tuleb igat värvi lambi osa. Punasel valgusel punane juhe, sinisel valgusel sinine juhe ning rohelisel valgusel roheline juhe.

Pildil nr. 29 kasutatud skeemi jaoks saame kirjutada koodi, mis sisaldab meil varem mainitud käsku #define, võimaldamaks pesadele nimesid anda, et programmi koostamine oleks mõningal määral lihtsam (praeguse koodi puhul ei ole see nii tähtis, sest kood on lühike, aga pikema koodi puhul teeb väljaviikudele nime andmine koodist aru saamise lihtsamaks). Samuti on kasutusel lokaalse muutujana „int“ tüüpi muutuja (lühend sõnast „integer“) ning on koostatud lisafunktsioon, millele viidatakse põhifunktsioonis.

Kood:

```
#define r_pin 3
#define s_pin 5
#define p_pin 6

void setup ()
{
pinMode (r_pin, OUTPUT); //siin anname Arduinole teada, milliseid pesasid kasutada. Nüüd on neil nimed, aga mitte numbrid (nimi muidugi viitab Arduino jaoks numbriga väljaviigule).
pinMode (s_pin, OUTPUT);
pinMode (p_pin, OUTPUT);
}

void loop()
{
varv (0,0,255);          //siin on korduses ainult viide funktsioonile nimega „varv“
                         ning sulgudes on selle funktsiooni muutujate väärtused, hetkel
                         peaks hakkama lamp põlema siniselt.
}

void varv (int punane, int roheline, int sinine){    //siin teeme värvid muutujateks, oluline on vaadata, et värvid ja väljaviigud oleks vastavuses (p_pin on ühendatud lambi punase osaga jne). Samuti oleneb muutujate järjestusest see, mis värv põlema läheb (täpsemalt on kirjas erinevate värvide tekitamise osas).
analogWrite(p_pin, punane);
analogWrite(r_pin, roheline);
analogWrite(s_pin, sinine);
}
```

Viimane osa „void varv“ on uus funktsioon, millele viitamine toimub põhifunktsiooni sees (umbes

samamoodi nagu käsu andmine). Antud juhul on meil tegemist funktsiooniga, kus on kolm muutujat. Nendeks on lampide värvid ning nende intensiivust muudetakse pulsilaiusmodulatsiooni abil, kasutades käsku: „analogWrite ()“ kus sulgudes on ära defineeritud väljaviik (näiteks „p_pin“ ning muutuja nimi ehk väärtus, mis muutub vastavalt põhifunktsioonis antud väärtustele. Näiteks, kui põhifunktsioonis on kirjas: „varv (0,0,255)“ siis saadetakse signaal täisvõimsusel (255 väärtust) sinise valguse välja saatmiseks. Kui näiteks kõik väärtused oleks 255, siis põleks valge valgus. Seega saab kolme põhivärvi intensiivsust kombineerides tekitada kõiki muid värve. Järgnevalt ongi ära toodud mõningate värvide saamiseks sobivad kombinatsioonid.

Punane – (255,0,0)	//sest funktsioonis „varv” on „int” tüüpi muutuja nimega „punane” esimene
Roheline – (0,255,0)	//sest funktsioonis „varv” on „int” tüüpi muutuja nimega „roheline” teine
Sinine – (0,0,255)	//sest funktsioonis „varv” on „int” tüüpi muutuja nimega „sinine” kolmas
Kollane – (255,255,0)	//sest punase ja rohelise lambi kombinatsioon annab kollase
Helesinine – (0,255,255)	//sest rohelise ja sinise lambi kombinatsioon annab helesinise
Lilla – (200,0,255)	//sest punase ja sinise kombinatsioon annab lilla (vähem punast teeb märgatavama värvi, aga igaüks saab ise proovida)
Roosa – (255,100,200)	//(piisaks ka ainult punasest ja sinisest, aga nii on värv selgemini eristatavam)
Oranž – (255,150,0)	//nagu kollane, aga punakas värv on tugevam
Valge – (255,255,255)	//kõik korraga annab kokku valge
Välja lülitamine – (0,0,0)	//kui voolu ei ole siis ei ole värvi ka

Sellist värvide jaotamist kasutatakse ka näiteks värvilistel ekraanidel erineva teabe näitamiseks. Asjadel, mis vajavad kohe tähelepanu on punane värv, mis vähem tähelepanu on oranž, veel vähem

tähelepanu on kollane ning taustainfo on valge. Kui kõik on korras, siis roheline.

Jadapordist käskude saatmine

Kuna jadapordis liiguvad andmed siiski kahes suunas, siis saame kasutada jadaporti ka käskude saatmiseks. Selleks saame kasutada juba varem koostatud skeemi pildilt nr 29. Meil on vaja ainult kood natuke ümber teha, õppida ära paar uut käsku ning kasutada näiteks stringi.

Uueks käsuks on „while“ tsükkel, see on tegevus, mida tehakse kogu aeg kui mingi tingimus on täidetud. Näiteks tegutseb kogu aeg, kui on piisavalt valge (saab valgusandurilt signaali) või kogu aeg, kui nupp on alla vajutatud. Tingimus on ära märgitud sulgudes ning kui see tingimus enam ei ole tõsi siis hakkatakse tegema muid asju.

Näide nupp

Selle tsükli näitena saame vaadata järgnevas koodis olevat osa, kus Arduino hakkab tegutsema ainult siis, kui jadapordi monitor on sisse lülitatud.

Pildil nr. 29 oleval skeemil paikneb mitmevärvilise dioodi juhtprogramm

```
String loetud;          //tekitame stringi nimega „loetud”, sellega hakkab Arduino asju võrdlema

# define p_pin 6        //siin anname käsuga #define väljaviikudele nimed
# define s_pin 5
# define r_pin 3

void setup ()           //ütleme Arduinole, mida peale hakata varem defineeritud pesadega ja
                        paneme tööle jadapordi monitori
{
 pinMode (p_pin, OUTPUT);
 pinMode (s_pin, OUTPUT);
 pinMode (r_pin, OUTPUT);
 Serial.begin (9600);
}

void loop ()                    //hakkab tööle programmi korduses olev osa
```

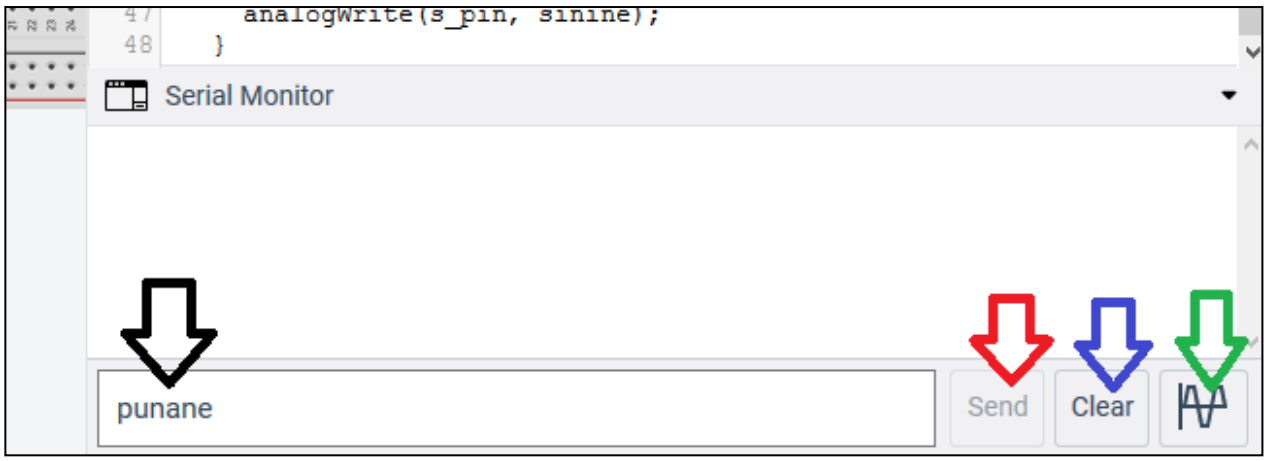
```
{
 while (Serial.available ())   //hakkab tööle „while" tsükkel, mis toimib siis, kui jadaport on
                               võimeline signaali vastu võtma
        {
 delay (10);                   //viiteaeg, et kasutaja saaks võtta näpud klaviatuurist eemale
 char c = Serial.read ();      //tekitatakse „char" tüüpi muutuja nimega „c", mille väärtust loetakse
                               jadapordi monitorist
 loetud+=c;                    //string „loetud" on nüüd „loetud+c", et Arduino oskaks jadapordi
                               monitori kirjutatut võrrelda
        }                      //„while" tsükli lõpp
if (loetud.length () > 0)      //„if" lause, milles seatakse tingimuseks, et stringi „loetud" pikkus{
                        oleks rohkem kui 0
 if (loetud == "punane"){      //„if" lause, mis kasutab funktsiooni „varv" punase valguse
                               tekitamiseks, kui jadapordi monitori on kirjutatud „punane"
 varv (255,0,0);
}
 if (loetud == "roheline"){    //sama mis eelmine, lihtsalt värv on seekord roheline
 varv (0,255,0);               //ära toodud numbrid tähendavad iga värvi lambi osa helendamise
                               intensiivsust
}
 if (loetud == "sinine"){      //sama mis eelmine, lihtsalt värv on seekord sinine
 varv (0,0,255);
 }
 if (loetud == "valge"){       //sama mis eelmine, lihtsalt värv on seekord valge
 varv (255,255,255);
}
 if (loetud == "eiole"){       //sama mis eelmine, lihtsalt seekord on lamp välja lülitatud
 varv (0,0,0);
}
 if (loetud == "h_sinine"){    //sama mis eelmine, lihtsalt värv on seekord helesinine
 varv (0,255,255);
}                              //lõpeb „if" lause, mis käsitleb helesinist valgust
```

```
  }                          //see lause käsib lõpetada jadaporti kirjutatud teksti pikkuse vaatamise
                             (esimene „if lause korduses)
 loetud = "";                //see paneb tühja koha jadapordi monitori kirjutamise osasse, et saaks
                             uut asja kirjutada
}                            //lõpeb kordus (void loop)

void varv (int punane, int roheline, int sinine)     //siin on funktsioon varv, millel on 3 muutujat,
{                                                    samasugune on varasemalt kirjeldatud lk. 3
  analogWrite (p_pin, punane);
  analogWrite (r_pin, roheline);
  analogWrite (s_pin, sinine);
 }
```

Jadapordist sisestamiseks on vaja vaadata ekraani alumist paremat osa ning vajutada seal „serial monitor", siis tekib selline kujutis (aga väiksem) nagu pildil nr. 30. Seal on musta noolega ära toodud koht. kuhu saab teksti sisestada (antud näites siis kirjutada „punane") ning nupp „send" (märgitud hetkel ära punase noolega). Viimase asemel saab kasutada ka klaviatuuril nuppu „enter". Jadapordi monitoril on Tinkercadi keskkonnas selline omadus, et ta jätab viimasena näidatud andmed veel ette ning seega oleks mõistlik peale seadete muutmist ning enne uute seadetega tegutsema hakkamist, vajutada nupule „Clear" (pildil on märgitud sinise noolega).

Samuti saab Tinkercadis jadaporti kasutades näidata andmete muutumist graafikuna, kasutades rohelise noolega märgitud nuppu. Sama on ka võimalik Arduinoga füüsilisel kujul, aga see vajab natuke rohkem seadistust.



Pilt nr. 30