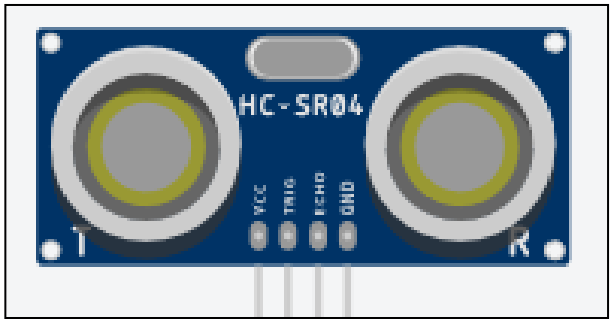
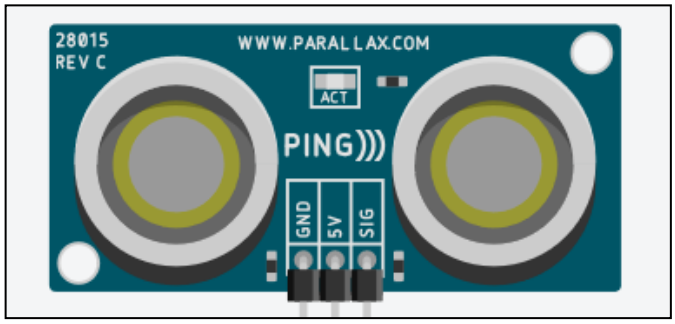
Arduino koolituse seitsmes tund – ultrahelianduri kasutamine

Õppematerjali koostajad: Indrek Karo ja Angela Leppik

Ultraheliandureid on mitut erinevat tüüpi. Tinkercadis on kasutusel kaks erinevat – nelja pistikuga (pilt nr. 46) ja kolme pistikuga (pilt nr. 47). Füüsilise Arduino puhul on tavaliselt kasutusel nelja pistikuga andur, sest see on odavam ning loogikast on lihtsam aru saada.



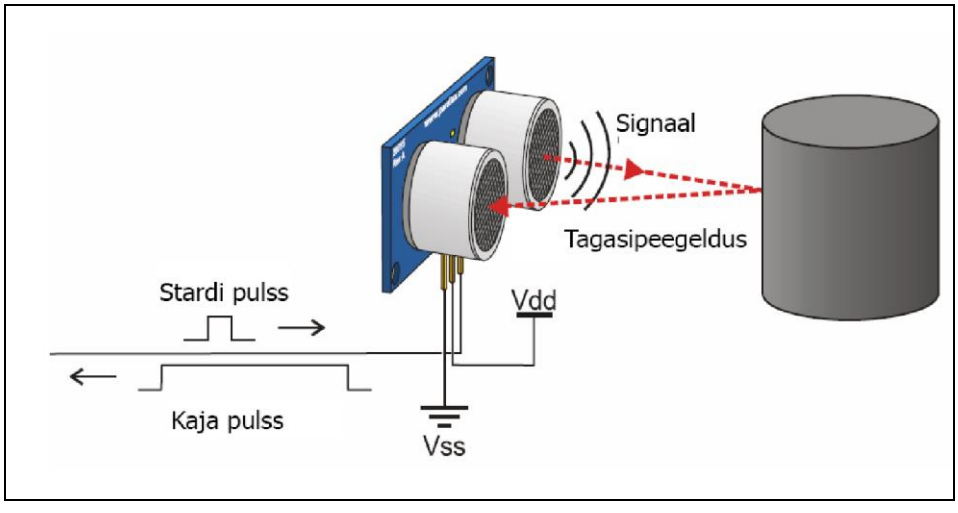
Pilt nr. 46



Pilt nr. 47

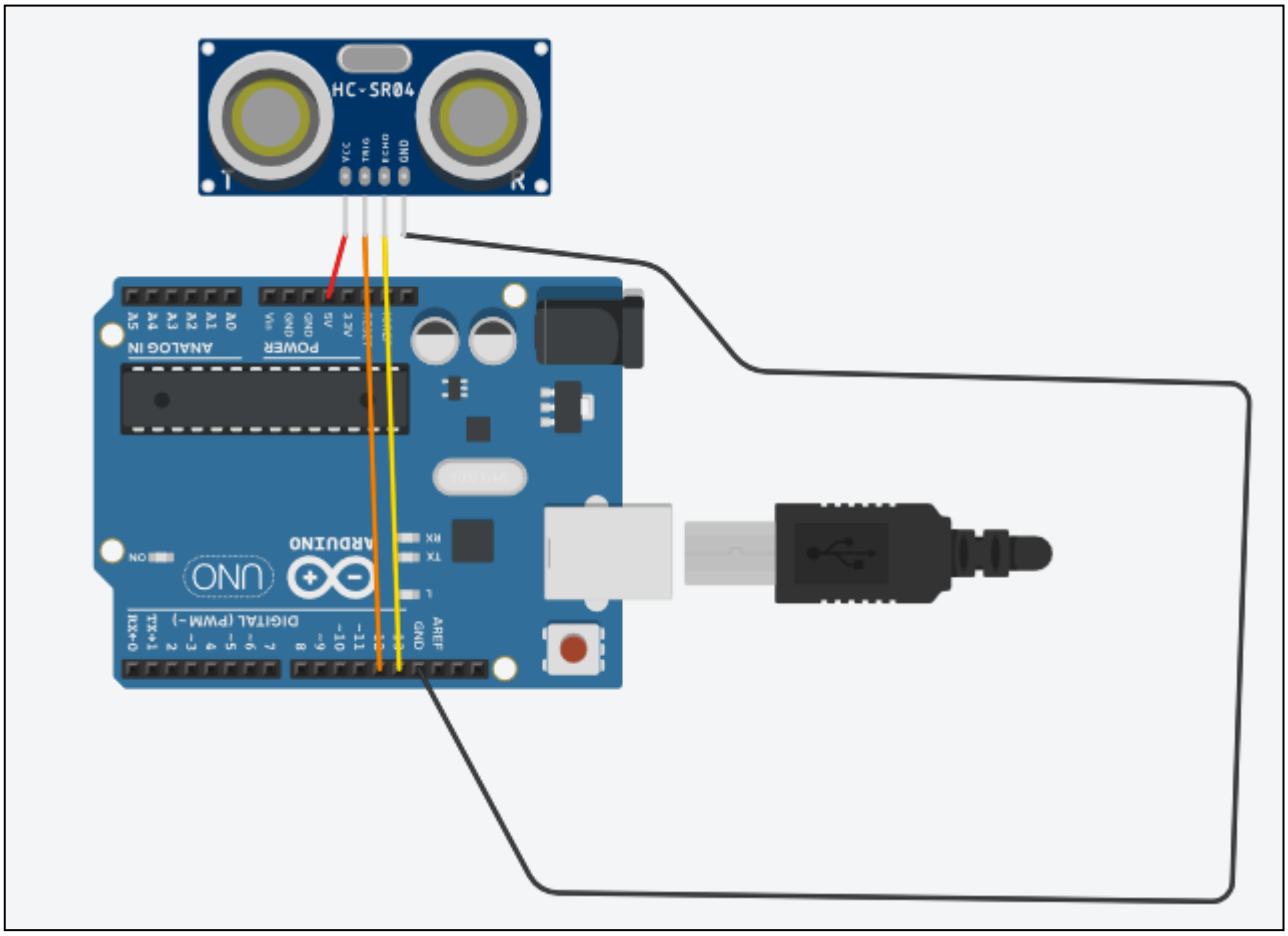
Ultrahelianduri loogika

Ultrahelianduri tööloogika on üsna lihtne – andur saadab heli välja ning ootab, kui see tagasi põrkab ning selle teabe alusel arvutab Arduino välja ees oleva takistuse kauguse. Ehk signaal saadetakse välja pesast TRIG ning tagasi tuleva signaali puhul saabub signaal, mille puhul muutub aktiivseks pesa ECHO. Kolme pistikuga anduril on tööloogika sama, aga signaali töötlemiseks kasutatakse ainult ühte pesa, nagu on näha pildil nr. 48. Ultraheli kasutavad ka erinevad loomad, näiteks nahkhiired ja delfiinid, sest nad tegutsevad tingimustes, kus nägemisest pole suurt kasu.



Pilt nr. 48

Kuna nelja jalaga andurid HC-SR04 on laialdasemalt kasutusel, siis on mõistlik kõigepealt vaadata kuidas ühendada ning luua kauguse mõõtmiseks sobivat programmi. Nende andurite teoreetiline mõõteulatus on 2-450 cm, praktiliselt aga mõõdavad 2–80 cm. Tinkercadi puhul see nii oluline ei ole. Enne kui teha programm, tuleks teada, kuhu midagi ühendada (pilt nr. 49).



Pilt nr. 49

Pildil nr. 49 näidatud anduri neli jalga on kasutusel järgnevalt:

VCC – 5 volti (ühendatakse Arduino pesasse 5V);

TRIG – välja saadetav signaal, ühendatakse digitaalsignaali pesasse (hetkel pesa nr. 12);

ECHO – vastu võetav signaal (see on varasemalt välja saadetud signaal, mis takistuselt tagasi põrkab), ühendatakse digitaalsignaali pesasse (hetkel pesa nr. 13);

GND – maandus, ühendatakse pesasse GND.

Pildil nr. 49 ära toodud skeemi programmis on kasutusel “long” tüüpi muutuja. See muutuja on kasutusel suurema mahuga tulemuste salvestamiseks ning salvestab kuni 32 bitti andmeid. Samuti

on programmis kasutusel valem, millega arvutatakse välja saadetud heli tagasi jõudmise aja abil anduri ees oleva takistuse kaugus.

Programmis on kasutatud pesade defineerimist, et oleks lihtsam meelde jätta ühendusi.

Samuti on kasutusel käsklus "pulseIn", mis põhimõtteliselt vaatab sissetuleva impulsi kestust mikrosekundites. Käsus on antud väljaviigu väärtus kas HIGH või LOW. Näiteks, kui on väljaviik nimega "kaja" ning väärtuseks HIGH siis seda kirjutatakse nii – pulseIn (kaja, HIGH). Arduino ootab kuni väljaviigu väärtus muutub LOW asemel HIGH ning hakkab aega mõõtma, kui väärtus on uuesti LOW, siis paneb aja mõõtmise kinni (nagu stopper). Selle käsuga võib tekkida probleeme täpsusega, kui impulss on liiga pika kestusega (toimib kõige paremini kui impulsi kestus on 10 mikrosekundit kuni 3 sekundit). Tavaliselt salvestatakse niiviisi mõõdetud aeg "long" tüüpi muutujana, et seda pärast edasi töödelda.

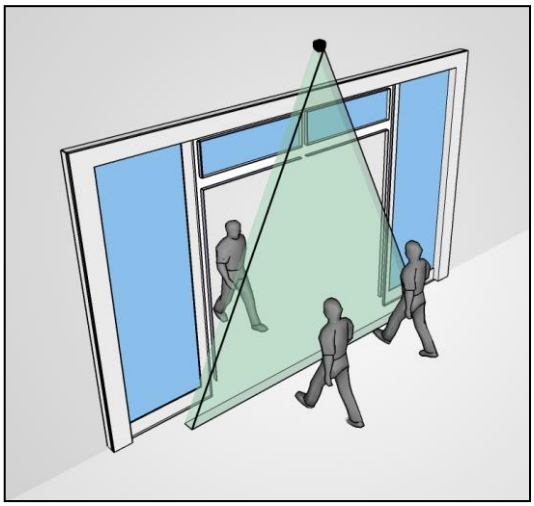
Ultraheli anduri programm

```
#define kisa 12                 //defineerime pesa signaali välja saatmiseks
#define kaja 13                 //defineerime pesa vastuvõetava signaali jaoks

void setup ()
{
Serial.begin (9600);            //paneme tööle jadapordi monitori kauguse näitamiseks
pinMode (kisa, OUTPUT);         //teatame Arduinole, et pesast "kisa" läheb signaal välja
pinMode (kaja, INPUT);          //teatame Arduinole, et tuleb oodata sisendit pesast "kaja"
}

void loop ()
{
long kestus, kaugus;            //tekitame kaks "long" tüüpi muutujat – "kestus" ja "kaugus"
                                //siin algab signaali saatmine – põhimõtteliselt saadetakse välja 10 mikrosekundiline impulss
  digitalWrite (kisa, LOW);     //"kisa" ei tee midagi
  delayMicroseconds (2);        //ootab 2 mikrosekundit
  digitalWrite (kisa, HIGH);    //pesast "kisa" läheb elekter välja
  delayMicroseconds (10);       //ootab 10 mikrosekundit
```

```
digitalWrite (kisa, LOW);          //“kisa” ei tee jälle midagi
                                   //kui kaua läheb aega signaali tagasi jõudmiseks?
kestus = pulseIn (kaja, HIGH);     //mõõdab impulsi kestust, et arvutada välja kaugust
kaugus = (kestus/2)/29,1;          //kaugus=aeg/2/29,1 – jagatakse kahega, sest peab arvestama
                                   heli edasi tagasi liikumist, heli kiirus on 29,1 cm
                                   mikrosekundis

Serial.print (kaugus);             //kauguse väljastamine
Serial.println ("cm");             //kauguse ühiku väljastamine
}
```

Inimeste loendaja

Kasutades ultraheliandurit ning eelnevat skeemi, saab ehitada inimeste loendamise masina. Põhimõtteliselt on tegemist lihtsalt programmiga, mis vaatab kauguse muutumist. Selle seletamiseks saab vaadata pilti nr, 50. Siin vaatab andur kaugusse. Saame määrata kauguse nimega B (näiteks 50 cm), kui miski anduri ette ilmub. Seda viimast kasutame kui muutujat B, mille abil saame võrrelda kauguse muutumist. Seega, iga kord kui keegi läheb andurile piisavalt lähedale (vähem kui B), siis Arduino arvab, et üks inimene on mööda läinud. Antud programmi puhul tuleb arvestada ka sellega, et inimestel võtab andurist möödumine natuke aega, seetõttu on kasutusel muutuja “mode”, mis arvestab inimeste möödumisega. Kuigi võib kasutada ka ainult käsku “delay”, kuid see ei arvesta eriti aeglaselt liikuvate inimestega.



Pilt nr. 50

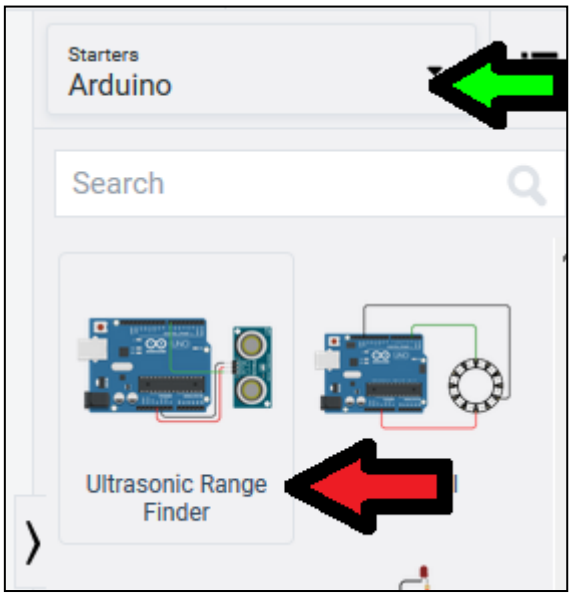
Inimeste lugeja programm

```
#define kisa 13                 //defineerime heli välja saatmise pesa
#define kaja 12                 //defineerime heli vastuvõtmise pesa
int B = 50;                     //"int" tüüpi muutuja, mõõdetava kauguse jaoks, millest väiksem arv
                                tähendab inimese möödumist
int loendur;                    //muutuja "loendur", see on loendatavate inimeste arv
int mode;                       //muutuja, millel on 2 erinevat väärtust: 0 – ootame inimest,
                                1 – ootame inimese möödumist

void setup ()
{
Serial.begin (9600);            //paneme tööle jadapordi monitori kauguse näitamiseks
pinMode (kisa, OUTPUT);         //teatame Arduinole, et pesast "kisa" läheb signaal välja
pinMode (kaja, INPUT);          //teatame Arduinole, et tuleb oodata sisendit pesast "kaja"
loendur = 0;                    //paneme paika muutuja "loendur" algväärtuse (0 – sest inimesi pole
                                veel mööda läinud
mode = 0;                       //anname algväärtuse muutujale "mode" (samuti 0, sest inimesi pole
                                veel mööda läinud)
}

void loop ()
{
long kestus, kaugus;            //siin hakkab kaugust mõõtma, kasutades "long" tüüpi muutujaid
  digitalWrite (kisa, LOW);     //siin algab signaali saatmine kauguse mõõtmiseks nagu varem
  delayMicroseconds (2);
  digitalWrite (kisa, HIGH);
  delayMicroseconds (10);
  digitalWrite (kisa, LOW);
  kestus = pulseIn (kaja, HIGH);
  kaugus = (kestus/2)/29,1;
```

```
if (mode == 0)                  //ootame inimest, kedagi ei jaluta anduri eest läbi
 {
  if (kaugus<B)                 //kui kaugus on väiksem, kui meie seatud piir (50 cm)
     {
    loendur++;                  //suurendame inimeste arvu
    mode = 1;                   //määrame, et ootame inimese ära minekut
     }
 }
 else                           //muud juhud
 {
  if (kaugus>B)                 //kui kaugus on suurem seatud piirist (50cm), siis kedagi ei jaluta
                                anduri eest läbi
  {
    Mode = 0;                   //siis on muutuja “mode” väärtus 0, et oodata järgmist inimest
  }
 }
 Serial.print (kaugus);         //näitab kaugust andurist
 Serial.print ("cm");           //kauguse ühik
 Serial.print ("\t");           //jätab tühiku vahele
 Serial.print (loendur);        //näitab möödunud isikude hulka
 Serial.println ("inimest");    //möödunud isikute ühik
 Delay (500);                   //ootab pool sekundit, et Arduino segadusse ei läheks ja ühte inimest
                                mitmeks inimeseks ei peaks
}
```

Ette antud skeemide kasutamine

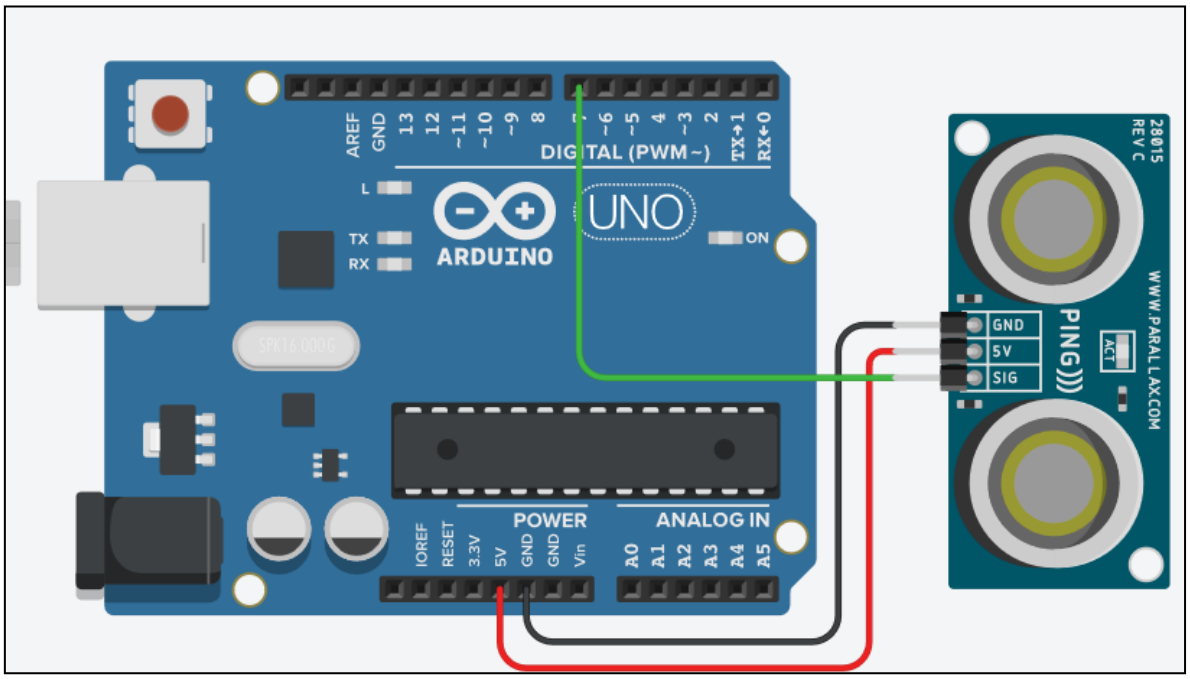
Tinkercad võimaldab kasutada ka varasemalt ette valmistatud skeeme. Selle näiteks saab tuua kolme pistikuga ultraheliandurи kasutamise, mis on ära toodud pildil nr. 51. Et kasutada Tinkercadi eelnevalt valmis tehtud skeemi, tuleb valida komponentide hulgast “Starters” (pildil ära märgitud rohelise noolega) ning nende hulgast Arduino. Siis tekivad komponentide aknasse erinevad ette valmistatud skeemid, mida saab valida. Sealt saab omakorda valida vajaliku skeemi, antud juhul ultraheliandurи kasutamise skeemi (pildil ära märgitud punase noolega). Need on varustatud kohe ka programmiga skeemi töötamiseks.



Pilt nr. 51

Kolme pistikuga ultraheliandur

Kolme pesaga ultraheliandurи kasutamise loogika on suhteliselt sarnane nelja pesaga anduriga. Lihtsalt tagasi põrkava signaali pikkust vaadatakse läbi ühe pesa (seda on näha ka pildil nr. 48). Programm on inglise keeles ning inglisekeelsete kommentaaridega, kuid arusaadav. Skeem ise on ära toodud pildil nr. 52.



Pilt nr. 52

Seega saab Tinkercadi kasutades teha natuke „sohki“ ning sellega säästa aega skeemide koostamise arvelt. Programmi loogika on selles, et saadetakse välja impulss, et alustada selle tagasi tuleku ootamist ning tagasi tuleva impulsi pikkus on proportsionaalne takistuse kaugusega andurist. Sellist tegevust nimetatakse ka pingimiseks ehk impulsi välja saatmiseks, et saada vastust.
Skeemil on 3 ühendust, mille kirjeldus on ära toodud ka kaasa antud programmis, mis on järgnevad:

5V – 5V ka Arduinol;
GND – maandus;
SIG – on signaal ning on ühendatud digitaalsignaali töötlemiseks väljaviiguga 7 Arduinol.

Selle kohta on ka Arduino kodulehel juhend: http://www.arduino.cc/en/Tutorial/Ping

Programmi selgitus

```
int inches = 0;          //muutuja, mis näitab kaugust tollides

int cm = 0;              //muutuja, mis näitab kaugust sentimeetrites

long readUltrasonicDistance (int triggerPin, int echoPin)  //“long“ tüüpi muutuja, mis omakorda
                                  sisaldab veel kahte muutujat, triggerPin (sama, mis “kisa”
                                  varasemalt) ning echoPin (sama, mis “kaja varasemalt), kuigi
                                  see muutuja meenutab oma olemuselt juba pisikest funktsiooni
{
  pinMode (triggerPin, OUTPUT);     //tutvustab signaali välja saatjat
  digitalWrite (triggerPin, LOW);   //ei saada signaali välja
  delayMicroseconds (2);            //ootab 2 mikrosekundit
  digitalWrite (triggerPin, HIGH);  //saadab signaali 10 mikrosekundi jooksul välja (järgmised 3
                                    rida)
  delayMicroseconds (10);
  digitalWrite (triggerPin, LOW);
  pinMode (echoPin, INPUT);         //loeb signaali muutujast echoPin ning annab helilaine
                                    liikumise aja mikrosekundites
  return pulseIn (echoPin, HIGH);   //käsuga “return” lõpetatakse ära funktsioon ning väljastatakse
                                    funktsiooni seest väärtus, kui seda on vaja
}
```

```
void setup ()
{
  Serial.begin (9600);                    //paneb tööle jadapordi monitori

}

void loop ()
{
  cm = 0.01723 * readUltrasonicDistance (7, 7);    //hakkab mõõtma pingimise kestust
                                                   sentimeetritega, kasutades selleks varem tutvustatud
                                                   muutuja “readUltrasonicDistance“) tulemust pesast 7
                                                   (kaks korda – välja mineva ja sisse tuleva signaaliga)
  inches = (cm/2.54);                     //muudab sentimeetrid tollideks
  Serial.print (inches);
  Serial.print ("in");
  Serial.print (cm);
  Serial.println ("cm");
  Delay (100);                            //ootab 100 millisekundit enne kui uuesti mõõdab
}
```

Antud juhul oli tegemist Tinkercadiga kaasa tuleva näidisprogrammiga, mis on lihtsalt üks võimalustest kasutada ultraheliandurit ning teha natukene „sohki“ skeemide koostamisel, sest tihtipeale on vaja teha keerukamaid skeeme ning on mugav, kui osa tööst on varem valmis tehtud.