Få-tiden-til-at-gå-[DIGITAL]-ur
==========
Copyright 2013, Richard Jørgensen.
Alle ophavsretlige rettigheder frafaldet 2015. (Kopier og brug som du har lyst.)

Forord:
=====
Denne vejledning var oprindelig lavet som fødselsdagsgave til en 11-årig dreng for lokke ham til at lære lidt om elektronik og programmering.
Det lykkedes!

Alt elektronik heri er købt via http://dx.com, og arduino udviklingssoftware er hentet fra http://arduino.cc.

I håbet om at andre også kan få noget ud af den frafalder jeg hermed alle de rettigheder jeg kan under dansk ophavsretslov.
Hvis nogen f.eks kan tjene penge på at udgive dette eller dele deraf som udervisningsmateriale er det med min fulde billigelse og glæde.
Jævnfør dansk ophavsretslov er der een paragraf jeg ikke kan frafalde, og som derfor stadig gælder:

> § 3. Ophavsmanden har krav på at blive navngivet i overensstemmelse med, hvad god skik kræver, såvel på eksemplarer af værket som når dette gøres tilgængeligt for almenheden.
> Stk. 2. Værket må ikke ændres eller gøres tilgængeligt for almenheden på en måde eller i en sammenhæng, der er krænkende for ophavsmandens litterære eller kunstneriske anseelse eller egenart.
>
> Stk. 3. Sin ret efter denne paragraf kan ophavsmanden ikke frafalde, medmindre det gælder en efter art og omfang afgrænset brug af værket.

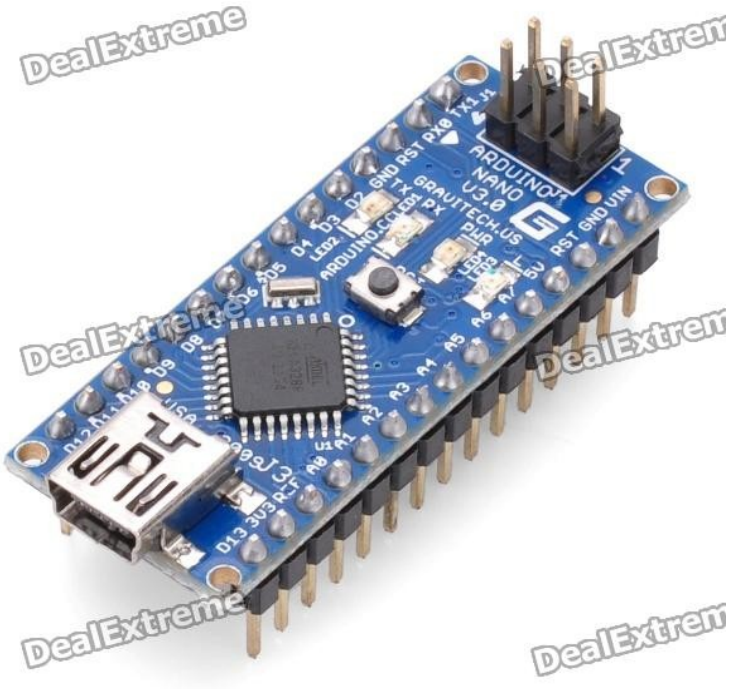
Jeg er naturligvis nysgerrig efter om andre kan bruge denne vejledning, så hvis du får glæde af den må du meget gerne sende mig en email om det.
Har du spørgsmål er du også velkommen til at kontakte mig.

Med venlig hilsen
Richard Jørgensen, (ric@haywire.dk)

Første program: Blink

Målet med denne opgave et at få lysdioden markeret "L" på Arduino
til at stå og blinke.

Sæt Arduino til computeren.
* Forbind Ardiono'en til en computer med USB
kablet
* Start Arduino udviklings program op
* Vælg Arduino board: Tools->Board->Arduino
Nano w/ATmega328
* Vælg seriel port: Tools->Seriel Port->COM???



Skriv det mindste gyldige program - et program
der gør ingenting:

```
void setup()
{
}

void loop()
{
}
```

* Tryk på "V" i øverste venstre hjørne for at
Verificere at programmet
  er korrekt.
  - Hvis computeren forstår programmert skriver den "Done Compiling" i
    bunden.
  - Hvis den ikke forstår det skriver den "Error Compiling" i bunden
    og tilføjer en halvkryptisk besked om hvad den ikke forstår.

* Tryk på "->" til højre for "V" for at lægge programmet over på
  Arduino kortet.

Forklaring:
Når arduino starter op forventer den at finde to funktioner ved navn
"setup" og "loop". Den kalder først "setup" een gang og derefter
kalder den "loop" igen og igen i det uendelige.

På Arduino kortet sidder lysdioden koblet på PIN 13. Så for at få den
til at blinke udvides ovenstående program således:

```
void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  digitalWrite(13, HIGH);
  delay(1000);
  digitalWrite(13, LOW);
  delay(1000);
}
```

* Tryk på "V" for at Verificere at programmet er korrekt.
* Tryk på "->" for at lægge programmet over på Arduino kortet.

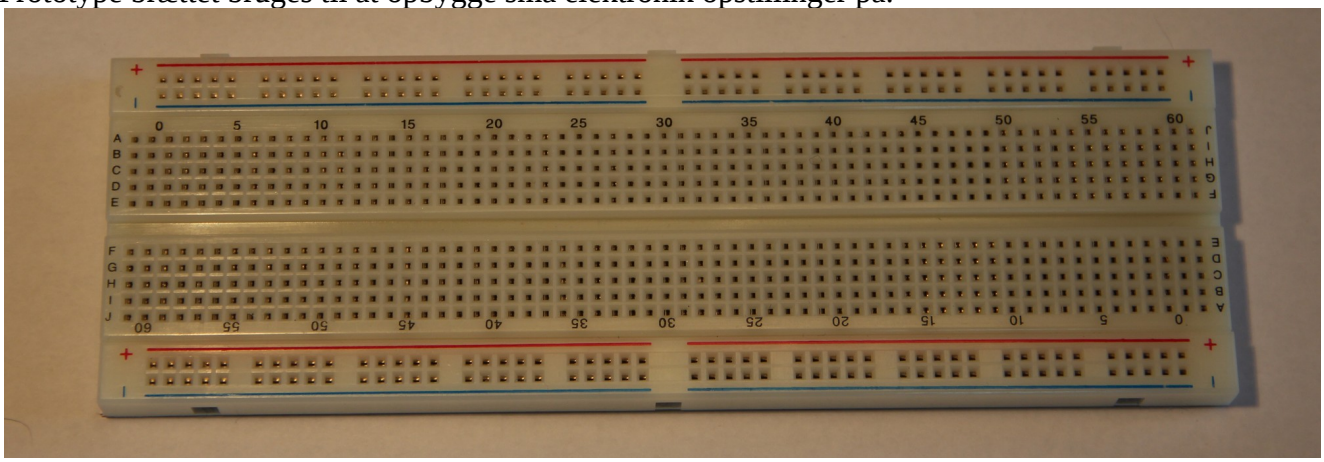
Hvis alt gik godt skulle lysdioden blinke nu.

Forklaring:

* pinMode(13, OUTPUT);
  PIN 13 kan enten bruges til at styre med "OUTPUT" eller måle med "INPUT".
  Vi skal styre en lysdiode så vi kalder Arduino-funktionen "pinMode"
  med argumenterne "(13, OUTPUT)" og afslutter med ";".
  Det skal sættes op før vi går igang med den egentlige opgave
  (blinke) så derfor hører den hjemme i "setup" funktionen.
* digitalWrite(13, HIGH); og digitalWrite(13, LOW);
  Arduino-funktionen "digitalWrite" sætter en OUTPUT PIN i "HIGH"
  (tændt, 5 volt spænding) eller "LOW" (slukket, 0 volt spænding).
* delay(1000);
  Funktionen "delay" venter et antal millisekunder. "delay(1000);"
  betyder "vent 1 sekund" (1000 millisekunder).
  Hvis du vil have lysdioden til at blinke hurtigere, så prøv med
  "delay(500);" eller "delay(100);".

Byg elektronik - styr en ekstern lysdiode

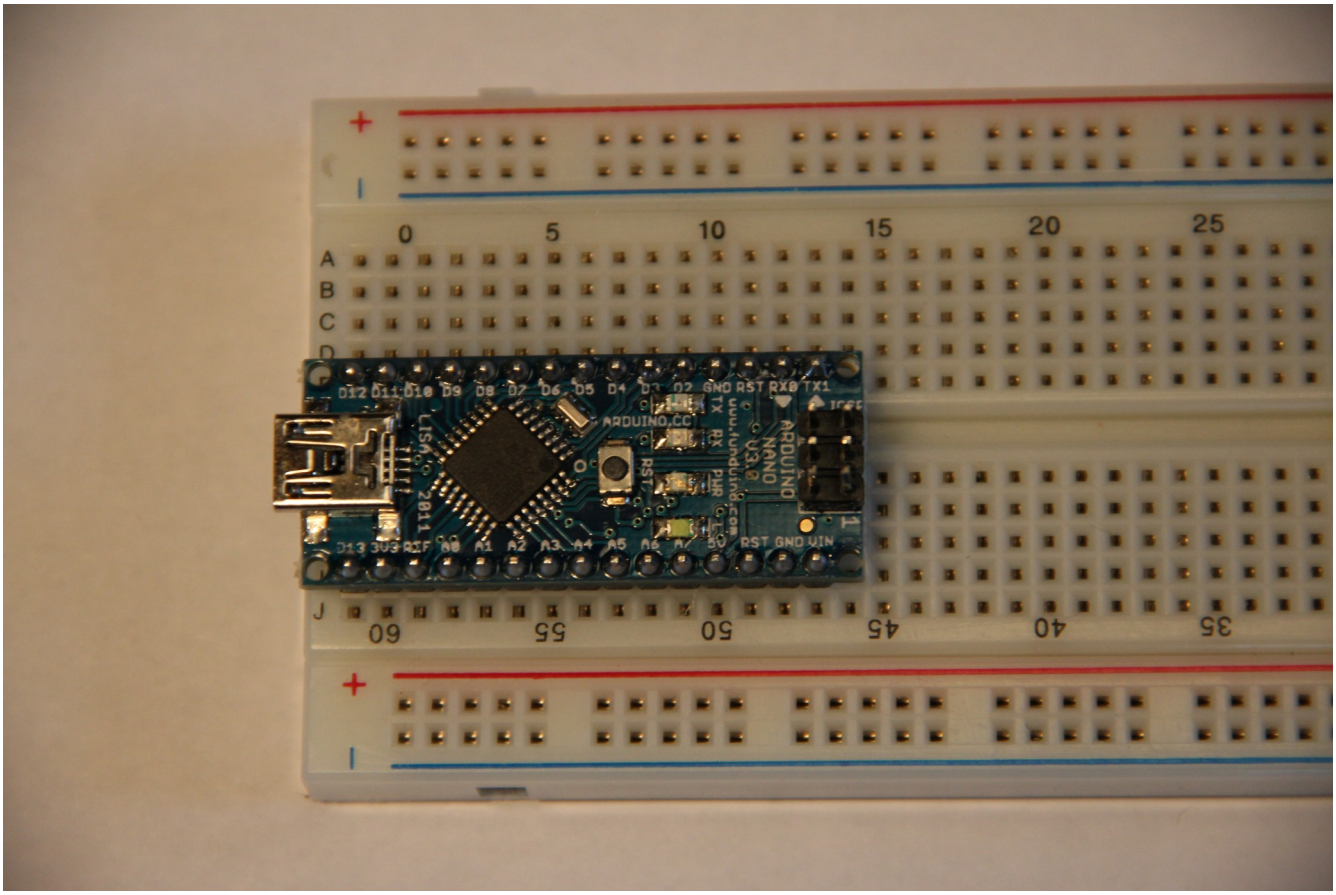
Prototype brættet bruges til at opbygge små elektronik opstillinger på.



I højre og venstre side to rækker huller forbundet på langs med brættet så man kan forbinde '+' og '-' i toppen, og dermed have nem adgang til '+' og '-' på brættet.

Resten af hullerne er forbundet på tværs, 5 ad gangen - A-B-C-D-E er forbundet og F-G-H-I-J er forbundet.

Tag strømmen fra Arduino. Når man bygger elektronik er der altid risiko for at man brænder elektronikken af hvis man laver en kortslutning eller forkert tilslutning.
Derfor slukker man altid for strømmen inden man ændrer en opstilling og dobbelt checker opstillingen inden man sætter strøm til igen.

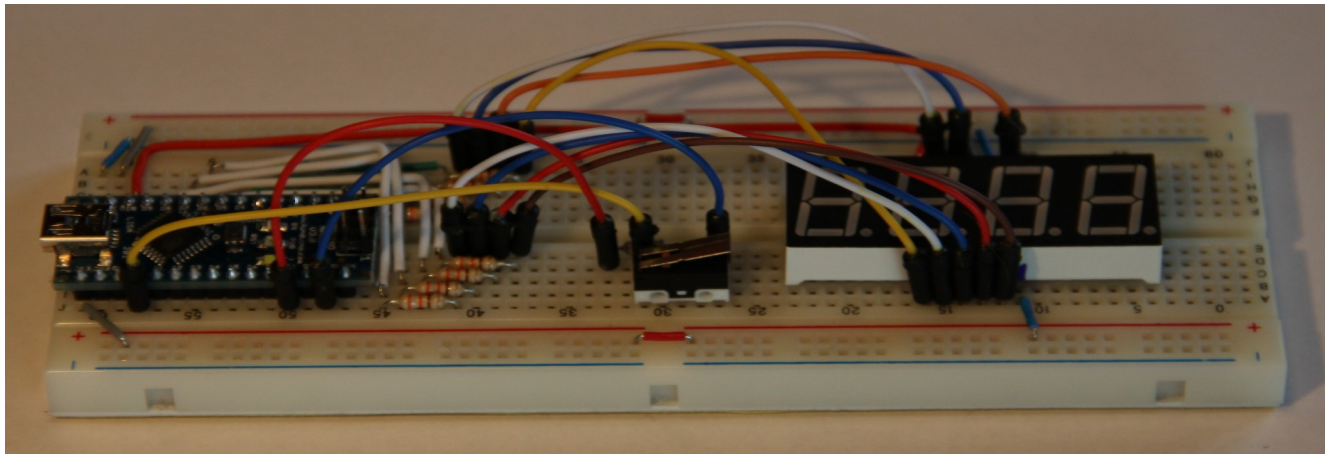


Lav forbindelse: [D13 -> modstand -> Lysdiode -> GND] således:

* Sæt Arduino i prototype brættet
* Sæt en ledning i brættet med den ene ende i samme række som Arduinos "D13" PIN og den anden ende i en ubrugt række.
* Sæt en modstand i prototype brættet med det ene ben i samme række som ledningens frie ende og det andet ben i en ny ubrugt række. Modstanden sættes i serie med lysdioden for at beskytte både den og Arduino mod der løber for meget strøm der kunne brænde elektronikken af.
* Sæt en lysdiode i brættet, med det lange ben i samme række som modstandens frie ende og det korte ben i en ny ubrugt række
* Sæt en ledning i brættet med den ene ende i samme række som lysdiodens frie ende og den anden ende i samme række som en af Arduinos "GND" PINs (der er een i hver side).
* Dobbelt check din opstilling - få meget gerne Henrik eller Richard til at checke den efter. Sæt derefter strøm på Arduino.

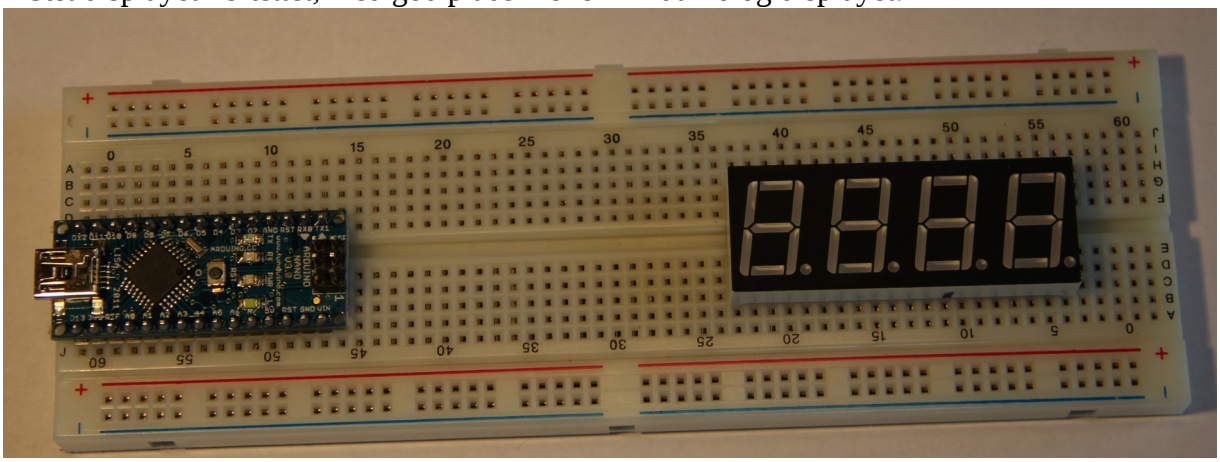
Den eksterne lysdiode skulle nu blinke i takt med lysdioden på Arduino.

Byg elektronik - Få lys i et firecifret display

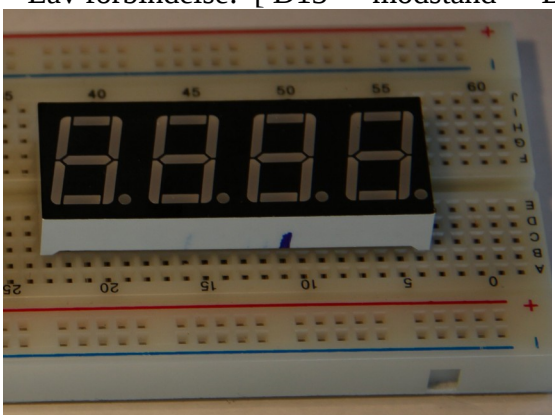


Det firecifrede display består af 32 lysdioder - hvert tal består af 7 linier og et punktum.
Displayet hat 12 ben de 8 ben (1,2,3,4,5,7,10,11) styrer de 7 linier og punktummet. De sidste 4 ben (6,8,9,12) (markeret med blå) styrer hvilket ciffer der tændes i.
Find den side med kun een blå streg - det er ben 1-6 og den blå streg er ud for ben 6.

* Sluk Arduino og fjern lysdioden.
* Sæt displayet i brættet, med god plads mellem Arduino og displayet.



* Lav forbindelse: [D13 -> modstand -> Ben-1] og [Ben-6 -> GND]



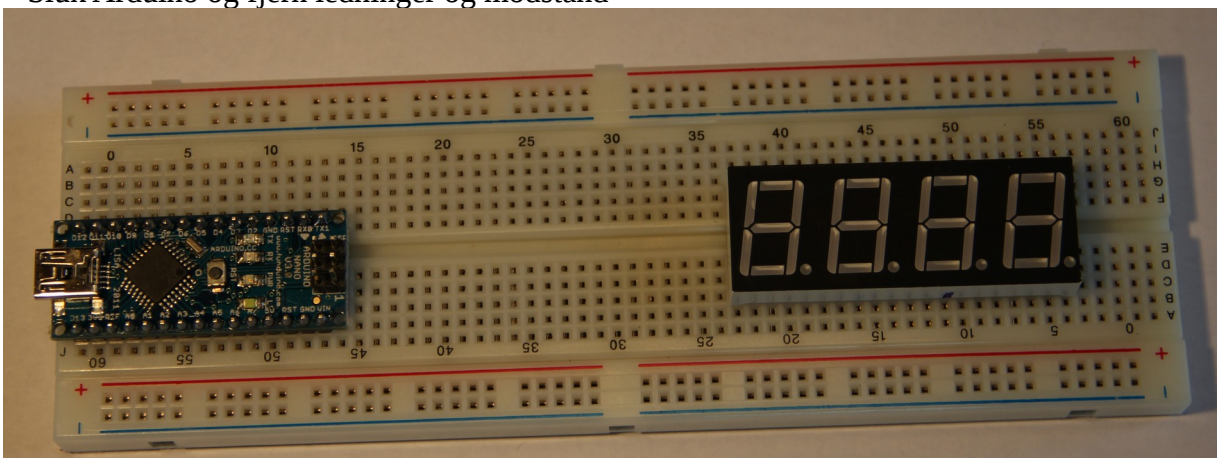
* Dobbelt check din opstilling - få meget gerne Henrik eller Richard
  til at checke den efter. Sæt derefter strøm på Arduino.
En af linierne i sidste ciffer af displayey skulle nu blinke i takt
med lysdioden på Arduino.

Hvis du er forsigtig kan du prøve at flytte ledningen fra Ben-1 over
til et af Ben 2-5 mens der er strøm på - de skulle flytte hvilken
linie i displayet der lyser.

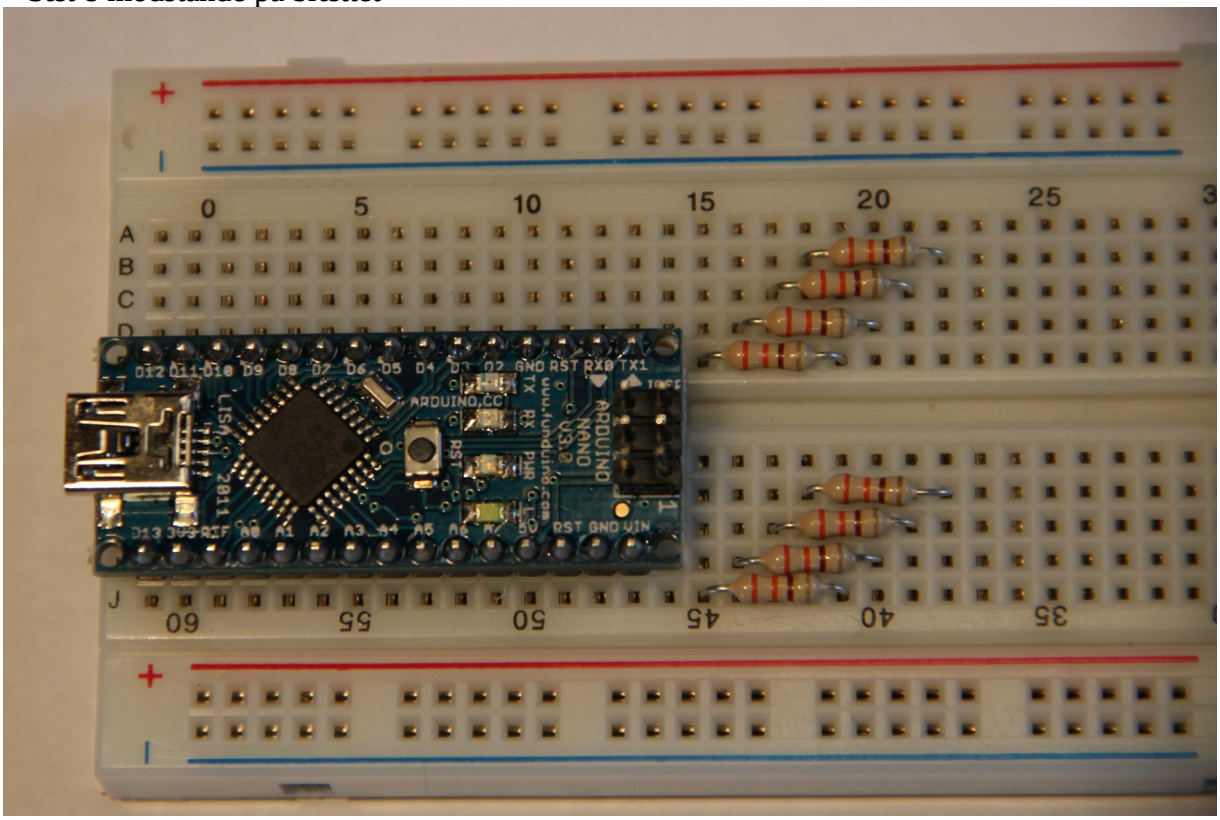
Byg elektronik - Styr et firecifret display

---

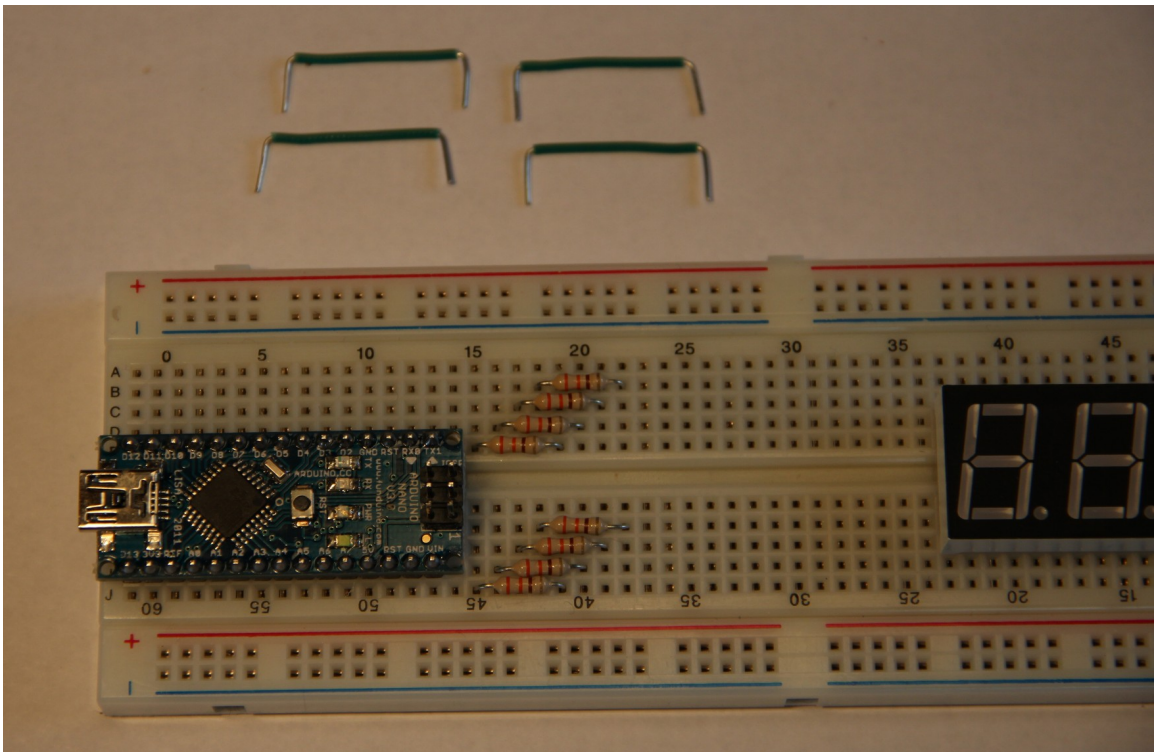
* Sluk Arduino og fjern ledninger og modstand

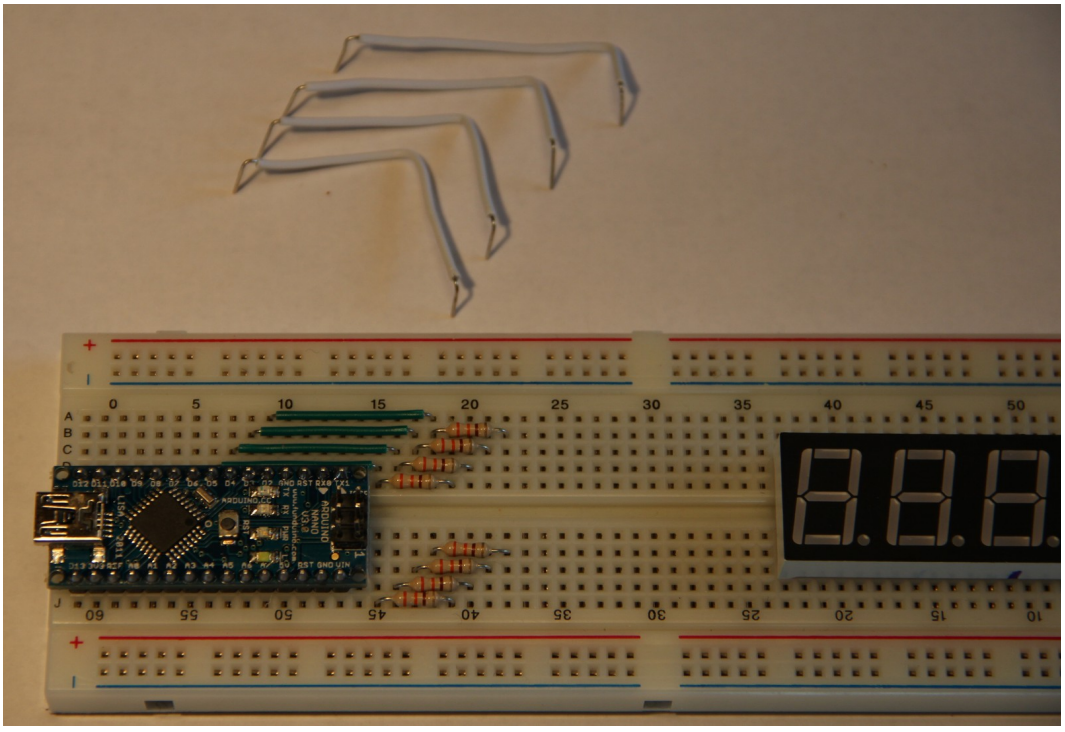


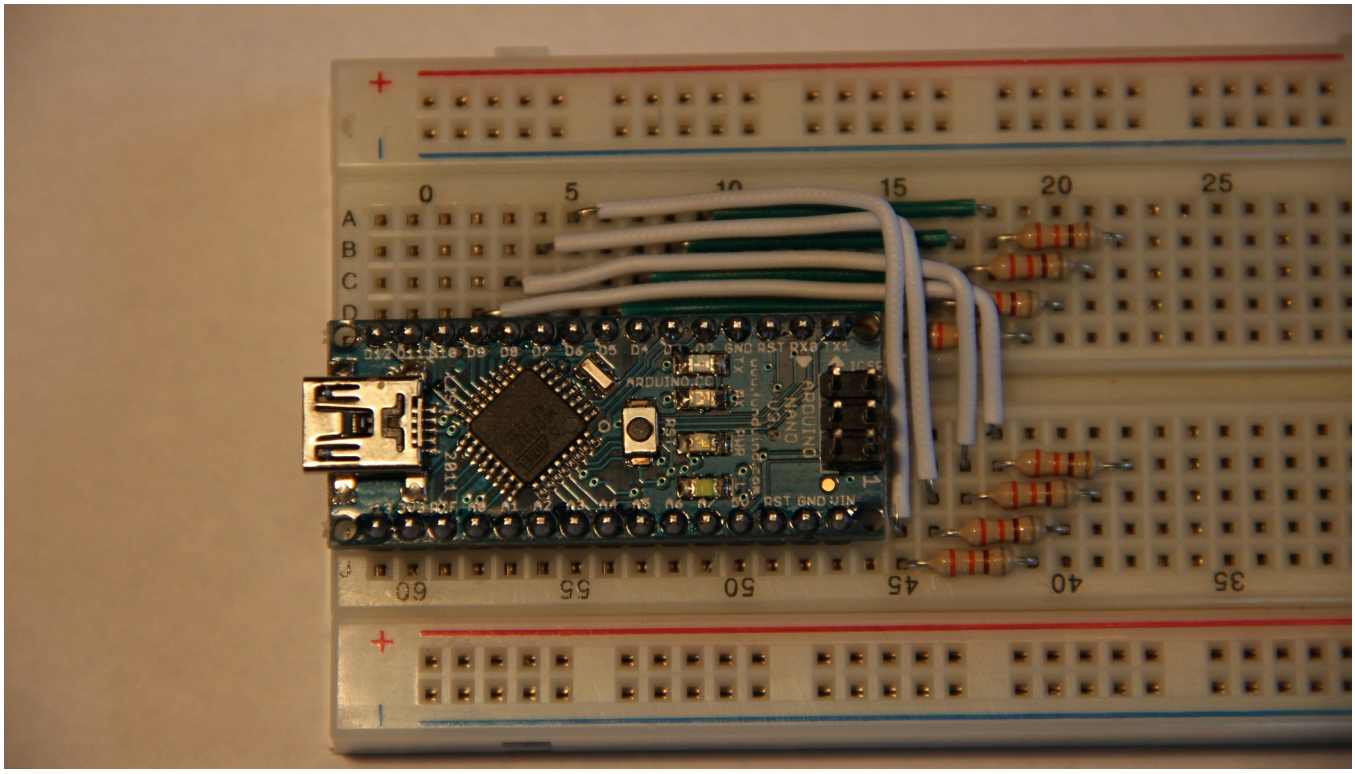
* Sæt 8 modstande på brættet



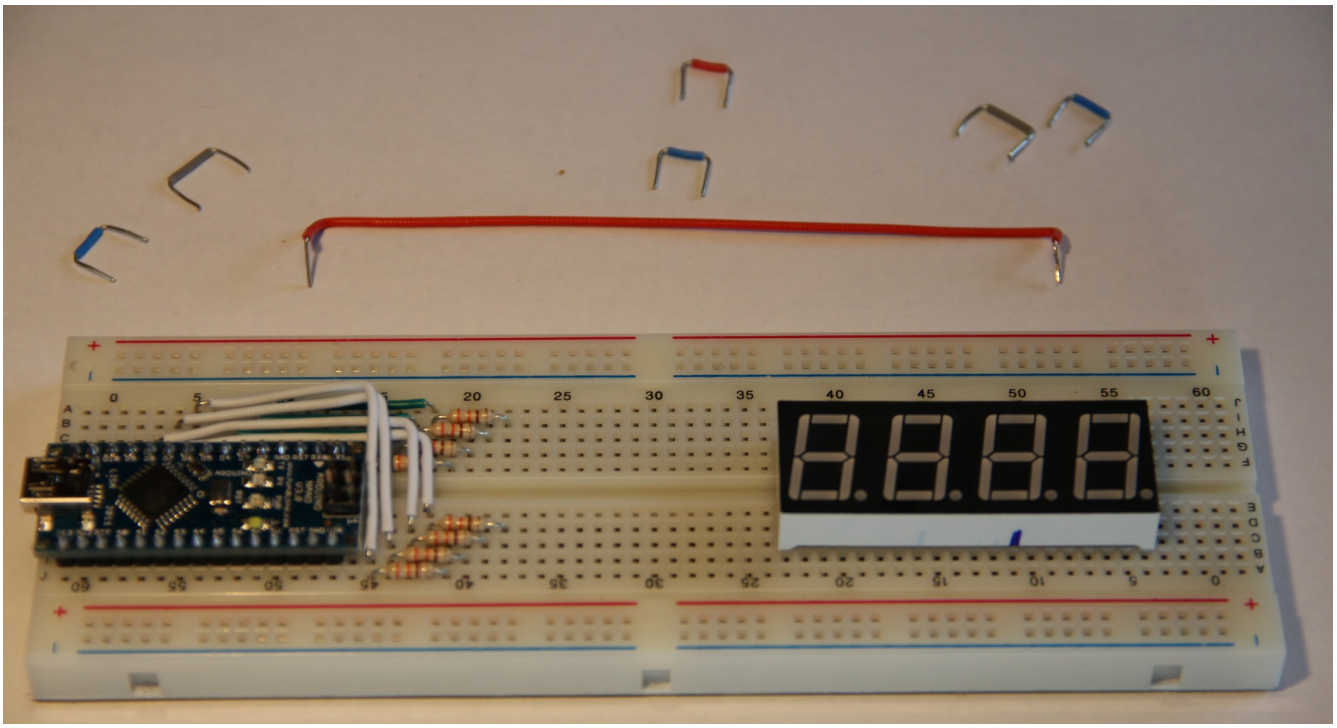
* Lav forbindelser fra digitale output til modstandene
  [D2 -> modstand]
  [D3 -> modstand]
  [D4 -> modstand]
  [D5 -> modstand]
  [D6 -> modstand]
  [D7 -> modstand]
  [D8 -> modstand]
  [D9 -> modstand]

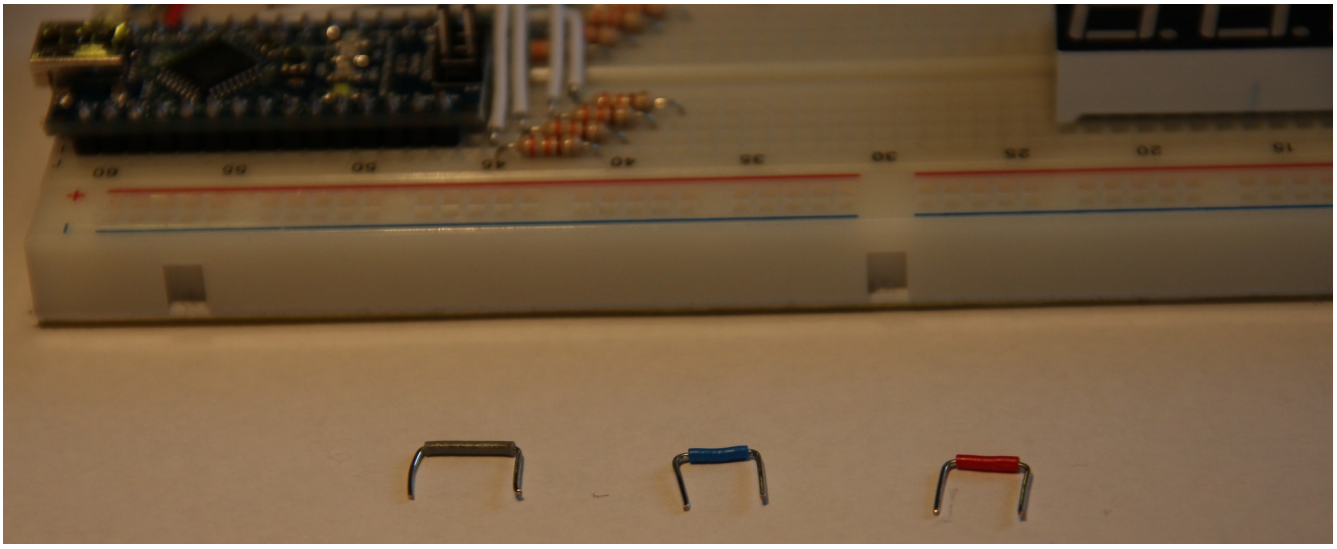


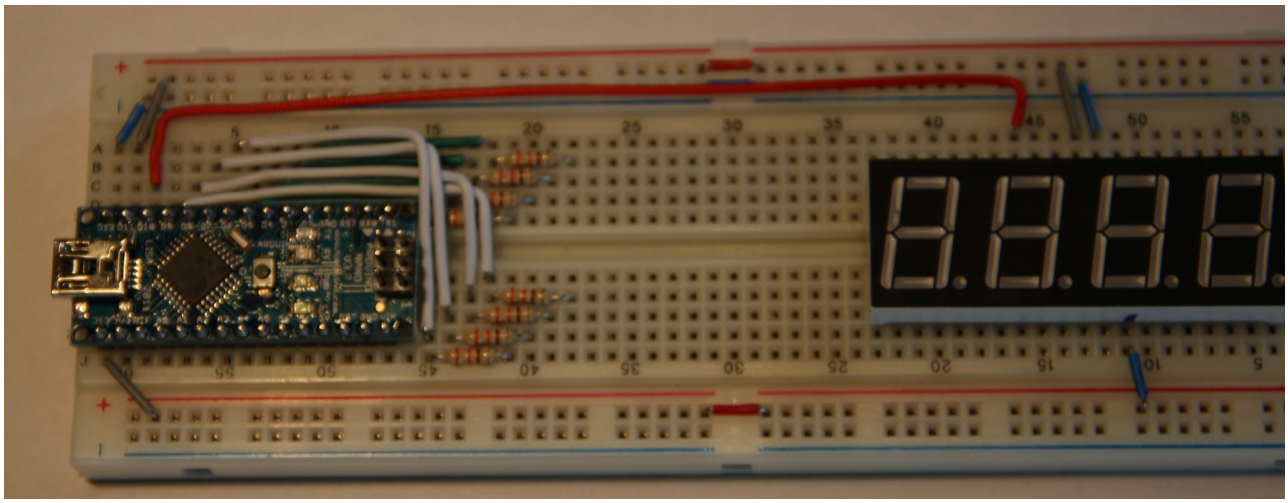


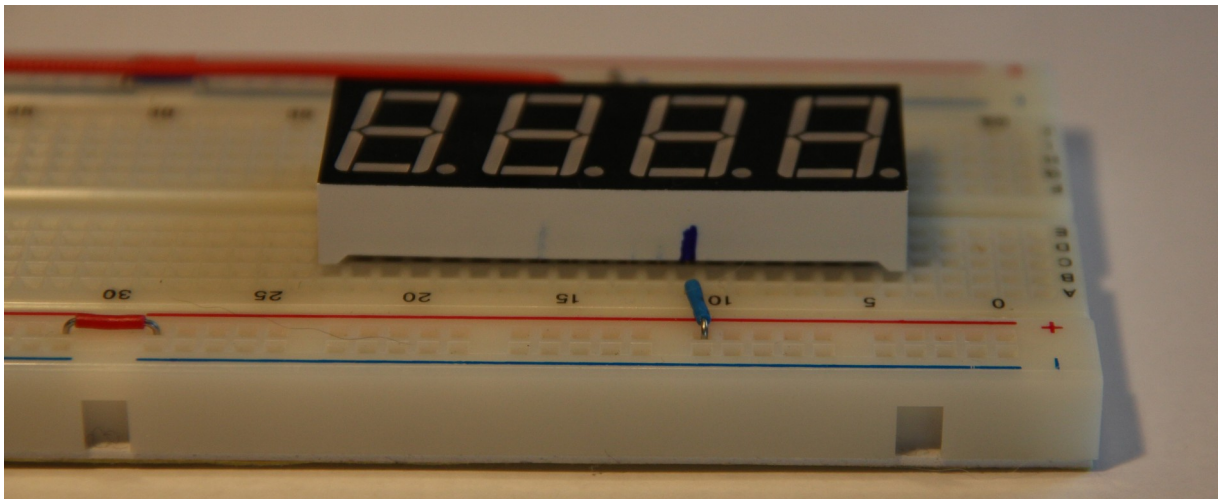

0
5
10
15
20
25
A
B
C
D
60
55
50
45
40
35
ARDUINO.CC
ARDUINO
NANO
RST GND VIN

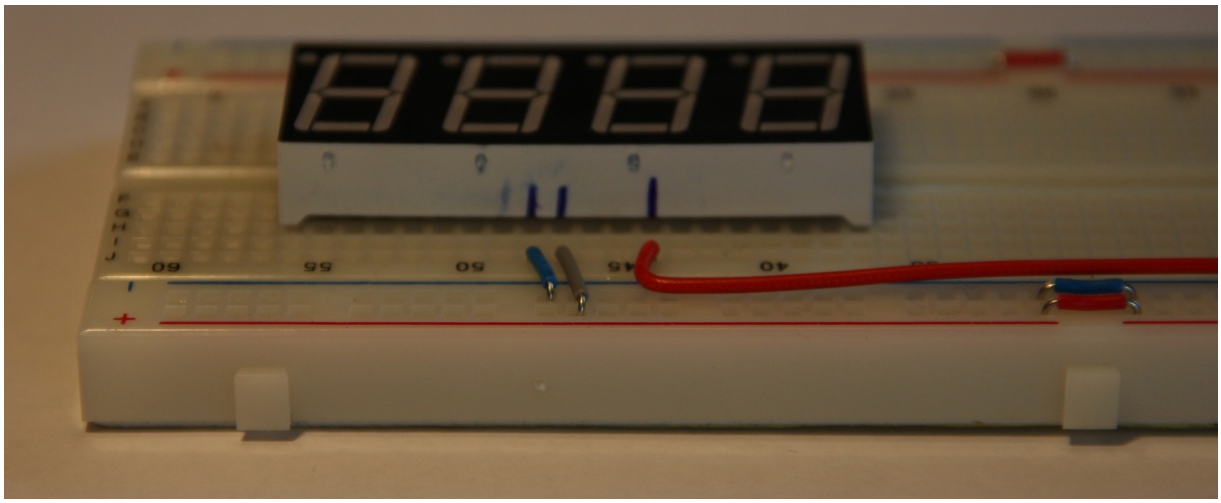
* Lav forbindelser fra digitale output direkte til de ben der er markeret med blå streg:
  [D13 -> Ben-6]
  [D10 -> Ben-8]
  [D11 -> Ben-9]
  [D12 -> Ben-12]



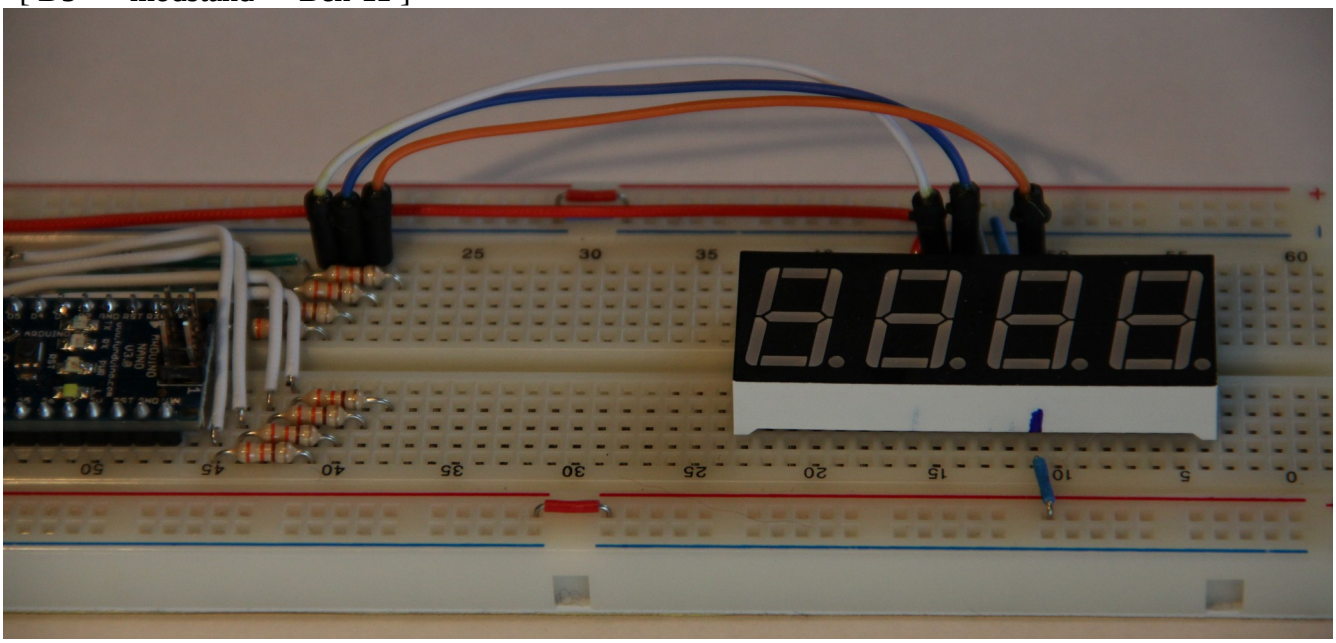


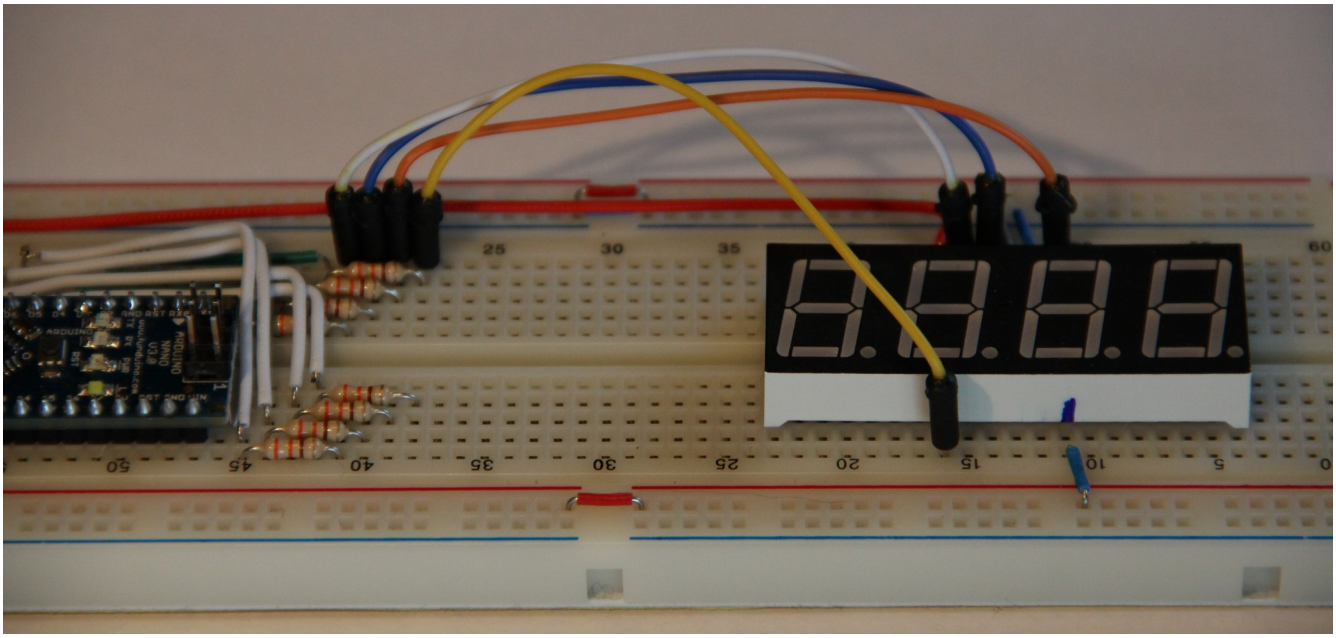


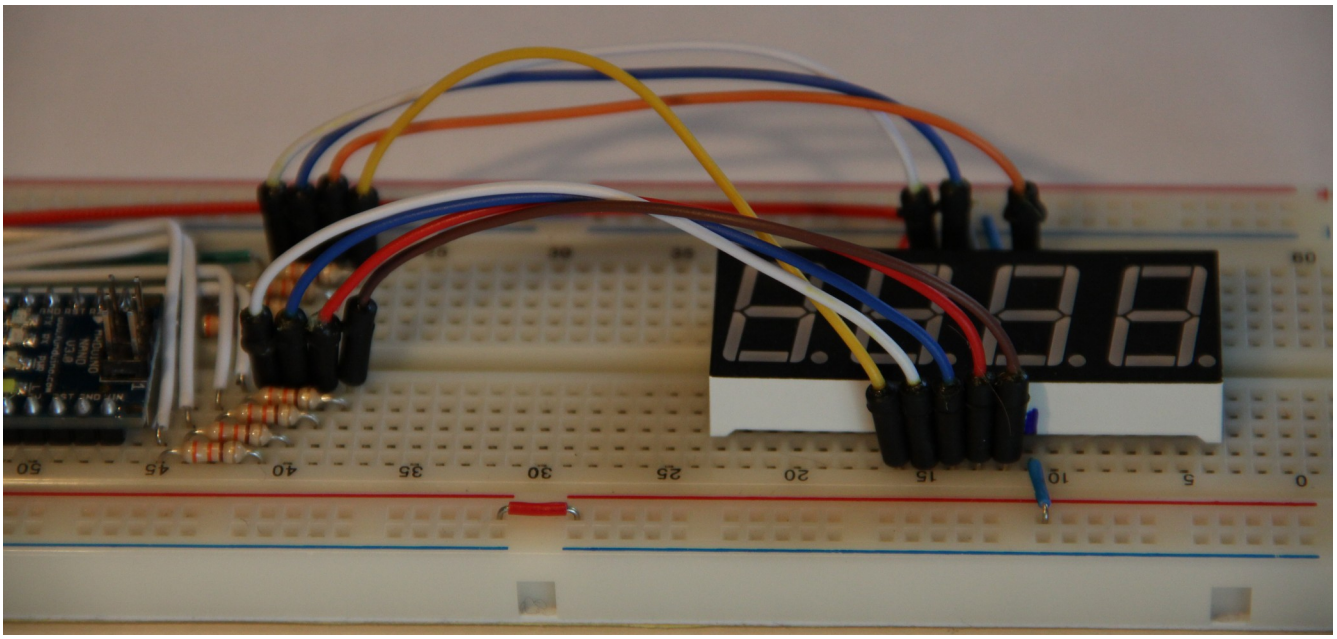


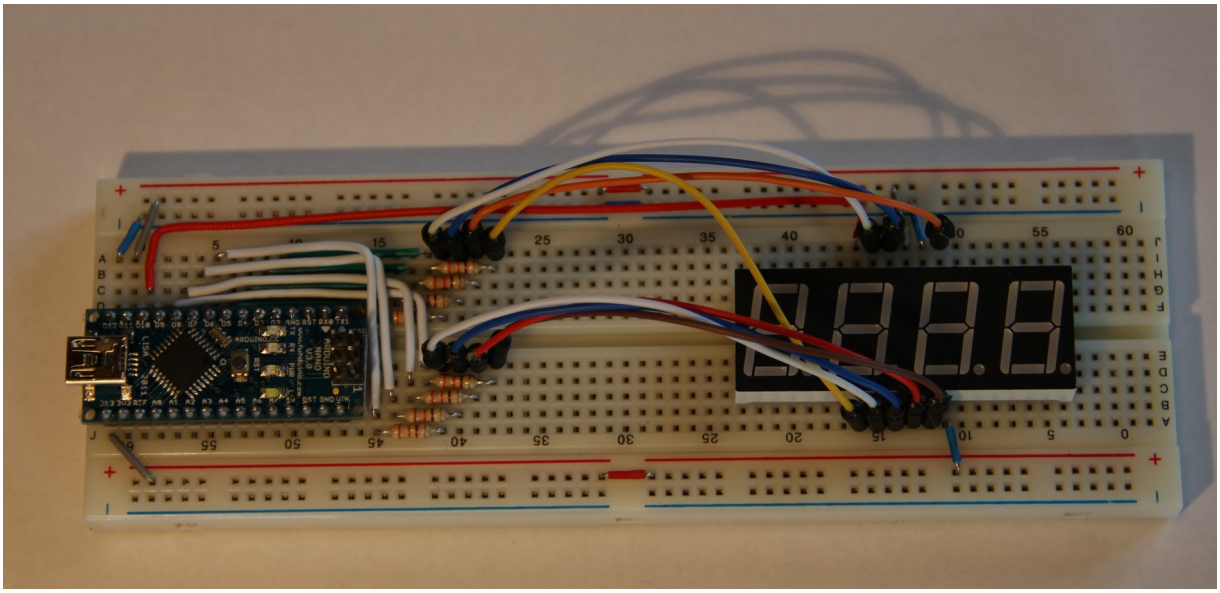


* Lav forbindelser fra digitale output, gennem modstande, til de ben der
  ikke er markeret med en blå streg:
  [D2 -> modstand -> Ben-1]
  [D6 -> modstand -> Ben-2]
  [D7 -> modstand -> Ben-3]
  [D8 -> modstand -> Ben-4]
  [D9 -> modstand -> Ben-5]
  [D3 -> modstand -> Ben-7]
  [D4 -> modstand -> Ben-10]
  [D5 -> modstand -> Ben-11]









Programmet til at blinke med alle lysdioderne ser således ud:

```
void setup()
{
  pinMode(2,  OUTPUT);
  pinMode(3,  OUTPUT);
  pinMode(4,  OUTPUT);
  pinMode(5,  OUTPUT);
  pinMode(6,  OUTPUT);
  pinMode(7,  OUTPUT);
  pinMode(8,  OUTPUT);
  pinMode(9,  OUTPUT);
  pinMode(10, OUTPUT);
  pinMode(11, OUTPUT);
  pinMode(12, OUTPUT);
  pinMode(13, OUTPUT);
  digitalWrite(10,  HIGH);
  digitalWrite(11,  HIGH);
  digitalWrite(12,  HIGH);
  digitalWrite(13,  HIGH);
}

void loop()
{
  digitalWrite(2, LOW);
  digitalWrite(3, LOW);
  digitalWrite(4, LOW);
  digitalWrite(5, LOW);
  digitalWrite(6, LOW);
  digitalWrite(7, LOW);
  digitalWrite(8, LOW);
  digitalWrite(9, LOW);
  delay(1000);
  digitalWrite(2, HIGH);
  digitalWrite(3, HIGH);
  digitalWrite(4, HIGH);
  digitalWrite(5, HIGH);
  digitalWrite(6, HIGH);
  digitalWrite(7, HIGH);
  digitalWrite(8, HIGH);
  digitalWrite(9, HIGH);
  delay(1000);
}
```

Check programmet med "V" og læg det over på Arduino med "->".

Løkker og variable

Programmet er efterhånden blevet temmelig langt - det kan skrives
simplere med en "for"-løkke.
Nedenstående program gør precis det samme:

```
void setup()
{
  for(int x=2; x<=13; x++) pinMode(x,  OUTPUT);
  for(int x=10; x<=13; x++)  digitalWrite(x,  HIGH);
}

void loop()
{
  for(int x=2; x<=9; x++) digitalWrite(x,  LOW);
  delay(1000);
  for(int x=2; x<=9; x++) digitalWrite(x,  HIGH);
  delay(1000);
}
```

Løkker og variable er to af de vigtigste begreber i programmering.

Forklaring på "for(int x=2; x<=13; x++) pinMode(x, OUTPUT);" :
* Variable:
  Du kender variable fra matematik i skolen hvor du har set opgaver
  som "3+x=5 Hvad er x?" eller "x=2 Løs regnestykket 3+x=?".
  Programmer ligner på det punkt matematik, men her er det computeren
  den skal sætte værdien af "x" ind og løse opgaven.
* for-løkken "for(int x=2; x<=13; x++)" har tre dele i sig, adskilt af ";".
 - "int x=2" betyder lav en variabel ved navn "x" og giv den værdien 2.
 - "x<=13" betyder "Er værdien af 'x' er mindre end eller lig med 13?"
 - "x++" betyder læg een til værdien af 'x'.

 Så længe "x<=13" er sandt udføres kommandoen "pinMode(x, OUTPUT);"
 bag for-løkken med den aktuelle værdi af 'x' hvorefter 'x' tælles een
 op og man tester igen "x<=13" osv.
 Resultatet er at "pinMode(x, OUTPUT);" bliver udført med alle værdier af
 'x' fra 2 til 13.

Kommentarer i programmer:

Efterhånden som programmer bliver mere komplekse er det en fordel at skrive kommentarer i programmet så man bedre kan overskue hvad det laver. Kommentarer har ingen effekt i programmet og markeres med to skråstreger: //
De to skråstreger og resten af linien bliver ignoreret af Arduino, så der kan man skrive hvad som helst.

Ovenstående program, nu med kommentarer:

```
void setup()
{
  // Sæt PIN D2-D13 til at være OUTPUT
  for(int x=2; x<=13; x++) pinMode(x,  OUTPUT);
  // Tænd alle fire cifre
  for(int x=10; x<=13; x++)  digitalWrite(x,  HIGH);
}

void loop()
{
  // Sluk alle streger i display'et
  for(int x=2; x<=9; x++) digitalWrite(x,  LOW);
  delay(1000);
  // Tænd alle streger i display'et
  for(int x=2; x<=9; x++) digitalWrite(x,  HIGH);
  delay(1000);
}
```

Styring af hvilket ciffer er tændt

Cifre styres af PIN 10, 11, 12 og 13, og tændes på HIGH mens de slukkes på LOW.

```
void setup()
{
  // Sæt PIN D2-D13 til at være OUTPUT
  for(int x=2; x<=13; x++) pinMode(x,  OUTPUT);
  // Tænd alle streger i display'et
  for(int x=2; x<=9; x++) digitalWrite(x,  LOW);
}

void loop()
{
  // Sluk alle fire cifre og tænd ciffer 1
  for(int x=10; x<=13; x++)  digitalWrite(x,  LOW);
  digitalWrite(10, HIGH);
  delay(1000);

  // Sluk alle fire cifre og tænd ciffer 2
  for(int x=10; x<=13; x++)  digitalWrite(x,  LOW);
  digitalWrite(11, HIGH);
  delay(1000);

  // Sluk alle fire cifre og tænd ciffer 3
  for(int x=10; x<=13; x++)  digitalWrite(x,  LOW);
  digitalWrite(12, HIGH);
  delay(1000);

  // Sluk alle fire cifre og tænd ciffer 4
  for(int x=10; x<=13; x++)  digitalWrite(x,  LOW);
  digitalWrite(13, HIGH);
  delay(1000);
}
```

Funktioner i programmer

Programmet er efterhånden blevet temmelig uoverskueligt, med meget gentagelse - det kan skrives simplere med en funktion.
Nedenstående program gør precis det samme:

```
void setup()
{
  // Sæt PIN D2-D13 til at være OUTPUT
  for(int x=2; x<=13; x++) pinMode(x,  OUTPUT);
  // Tænd alle streger i display'et
  for(int x=2; x<=9; x++) digitalWrite(x,  LOW);
}

void ciffer(int D)
{
  // Sluk alle fire cifre og tænd ciffer D
  for(int x=10; x<=13; x++)  digitalWrite(x,  LOW);
  digitalWrite(D, HIGH);
  delay(1000);
}

void loop()
{
  ciffer(10);
  ciffer(11);
  ciffer(12);
  ciffer(13);
}
```

Forklaring:
Ved at definere funktionen "void ciffer(int D)" udvides programmeringssproget med en ny kommando ved navn "ciffer". Funktionen er defineret med en variabel "int D". Når man bruger "ciffer" skal man give et tal med - f.eks "ciffer(3);". Det tal bliver bliver lagt ned i "D" og kan bruges inden i ciffer funktionen.

Visning af tallene 1 og 2

Det er på tide at få nogle tal i display'et istedet for bare at tænde alt:

```
void setup()
{
  // Sæt PIN D2-D13 til at være OUTPUT
  for(int x=2; x<=13; x++) pinMode(x,  OUTPUT);
}

void ciffer(int D)
{
     // Tænd Ciffer D
  digitalWrite(D, HIGH);
  delay(1000);
}

void tegn(int tal, int punktum)
{
  // Navngiv de PINs der svarer til segmenter i databladet
  int E=2, D=6, DP=7, C=8, G=9, B=3, F=4, A=5;

  // Sluk alle fire cifre
  for(int x=10; x<=13; x++)  digitalWrite(x,  LOW);

  // Sluk alle streger i display'et
  for(int x=2; x<=9; x++) digitalWrite(x,  HIGH);

  // Tænd eventuelt punktummet
     digitalWrite(DP, punktum);

  if(tal == 1)
  {
    digitalWrite(B, LOW);
    digitalWrite(C, LOW);
  }

  if(tal == 2)
  {
    digitalWrite(A, LOW);
    digitalWrite(B, LOW);
    digitalWrite(D, LOW);
    digitalWrite(E, LOW);
    digitalWrite(G, LOW);
  }
}
```

```
void loop()
{
  tegn(1, LOW);
  ciffer(10);

  tegn(2, HIGH);
  ciffer(11);

  tegn(3, LOW);
  ciffer(12);

  tegn(4, LOW);
  ciffer(13);
}
```

Forklaring:
Funktionen "tegn" indeholder en ny konstruktion: "if(tal == 1) { ... }".
Det betyder: hvis variablen 'tal' har værdien 1 så udfør
kommandoerne mellem '{' og '}', ellers spring over dem.
Bemærk: Dobbelte lighedstegn '==' bruges til at sammenligne to værdier, mens
enkelt lighedstegn '=' bruges til at sætte en variabel til en værdi.

Der mangler definitioner af hvordan tallene 3 og 4 tegnes, så de
sidste to cifre er blanke.
Kan du med ved hjælp af databladet udvide "tegn" funktionen til at
vise 3 og 4 også? Og hvad med 5, 6, 7, 8, 9 og 0?

Visning af fire forskellige cifre på een gang

Sidste program viste 4 tal på hvert sit ciffer, eet ad gangen. Vi vil
gerne have de fire tal vist samtidigt.
Hemmeligheden er at skifte så hurtigt mellem de fire cifre at øjet
ikke kan nå at følge med.
Kommandoen "delay(1000)" i "ciffer" funktionen betyder at hvert ciffer
vises i 1000 millisekunder.
Hvordan ser det ud hvis du ændrer den til "delay(100)"? Hvad med "delay(10)"?
Eller "delay(1)"?

Få tiden til at gå

Ligesom ved et analogt ur skal vi have fat i et "pendul" for at tælle sekunder. I elektronik bruger man et krystal hvor man elektrisk kan måle svingningerne.
Arduino'ens krystal er den lille sølvfarvede komponent på størrelse med et riskorn. Den svinger med 16Mhz - dvs 16 millioner gange i sekundet.
Der er nogen der har udvidet programmeringssproget med nogle funktioner der gør det nemt at få fat i krystallens svingninger.
For at benytte dem, skriver man følgende to linier i toppen af programmet:

```
#include <SimpleTimer.h>
SimpleTimer timer;
```

Under "setup" sætter man hvilken funktion der skal kaldes, og hvor ofte:
(Hvis du vil se uret opdatere uden at vente et døgn så sæt opdateringen til 60 istedet for 60000 - så tager et døgn ca. halvandet minut)

```
timer.setInterval(60000, pendul);
```

så laver man den funktion der skal kaldes:

```
void pendul()
  {
   ...
  }
```

Og i loop kalder man timer.run() der checker om pendul() skal kaldes nu.

```
void loop()
{
    timer.run();
    ...
}
```

Derudover har jeg tilføjet variable i toppen til at holde tiden i, som bliver talt op i "pendul" og brugt i "loop"
(Du kan stille uret ved at ændre start værdierne her)

```
// Lav variabler for tiden og sæt klokken til 11:59.
// 'Tid' tæller minutter siden midnat
int tid=60*11+59;
int time2, time1, minut2, minut1;
```

Ved beregninger med heltal på en computer betyder division at man smider alt efter kommaet væk, så ved "19/10" der burde give "1,9" smider man de "0,9" væk og resulatet er at "19/10 = 1".
Det udnyttes til at beregne timer og minutter i "pendul".

Så det endelige program ser sådan her ud:

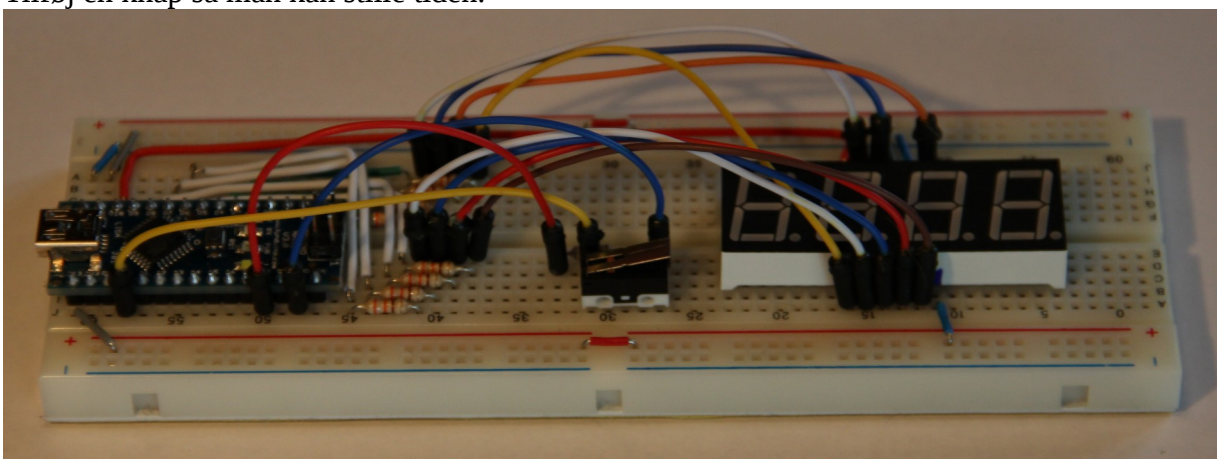
```
#include <SimpleTimer.h>
SimpleTimer timer;

// Lav variabler for tiden og sæt klokken til 11:58.
// 'Tid' tæller minutter siden midnat
int tid=60*11+58;
int time2, time1, minut2, minut1;

void setup()
{
  // Sæt PIN D2-D13 til at være OUTPUT
  for(int x=2; x<=13; x++) pinMode(x,  OUTPUT);

  // Kald funktionen "pendul" hvert minut (60000 millisekunder)
  timer.setInterval(60000, pendul);
  pendul();
}

void pendul()
{
  tid++;

  // Der er 1440 minutter på en dag
  if(tid >= 1440)
  {
    tid = tid - 1440;
  }
  time2 = tid/600;
  time1 = tid/60 - 10*time2;
  minut2 = tid/10 - 6*(10*time2 + time1);
  minut1 = tid - 10*minut2 - 60*(10*time2 + time1);
}

void ciffer(int D)
{
  // Tænd Ciffer D
  digitalWrite(D, HIGH);
  delay(1);
}
```

```
void tegn(int tal, int punktum)
{
  // Navngiv de PINs der svarer til segmenter i databladet
  int E=2, D=6, DP=7, C=8, G=9, B=3, F=4, A=5;

  // Sluk alle fire cifre
  for(int x=10; x<=13; x++)  digitalWrite(x,  LOW);

  // Sluk alle streger i display'et
  for(int x=2; x<=9; x++) digitalWrite(x,  HIGH);

  // Tænd eventuelt punktummet
  digitalWrite(DP, punktum);

  if(tal == 1)
  {
    digitalWrite(B, LOW);
    digitalWrite(C, LOW);
  }

  if(tal == 2)
  {
    digitalWrite(A, LOW);
    digitalWrite(B, LOW);
    digitalWrite(D, LOW);
    digitalWrite(E, LOW);
    digitalWrite(G, LOW);
  }

  if(tal == 3)
  {
    digitalWrite(A, LOW);
    digitalWrite(B, LOW);
    digitalWrite(C, LOW);
    digitalWrite(D, LOW);
    digitalWrite(G, LOW);
  }

  if(tal == 4)
  {
    digitalWrite(B, LOW);
    digitalWrite(C, LOW);
    digitalWrite(F, LOW);
    digitalWrite(G, LOW);
  }

  if(tal == 5)
  {
    digitalWrite(A, LOW);
```

```
    digitalWrite(C, LOW);
    digitalWrite(D, LOW);
    digitalWrite(F, LOW);
    digitalWrite(G, LOW);
  }

  if(tal == 6)
  {
    digitalWrite(A, LOW);
    digitalWrite(C, LOW);
    digitalWrite(D, LOW);
    digitalWrite(E, LOW);
    digitalWrite(F, LOW);
    digitalWrite(G, LOW);
  }

  if(tal == 7)
  {
    digitalWrite(A, LOW);
    digitalWrite(B, LOW);
    digitalWrite(C, LOW);
  }

  if(tal == 8)
  {
    digitalWrite(A, LOW);
    digitalWrite(B, LOW);
    digitalWrite(C, LOW);
    digitalWrite(D, LOW);
    digitalWrite(E, LOW);
    digitalWrite(F, LOW);
    digitalWrite(G, LOW);
  }

  if(tal == 9)
  {
    digitalWrite(A, LOW);
    digitalWrite(B, LOW);
    digitalWrite(C, LOW);
    digitalWrite(D, LOW);
    digitalWrite(F, LOW);
    digitalWrite(G, LOW);
  }

  if(tal == 0)
  {
    digitalWrite(A, LOW);
    digitalWrite(B, LOW);
    digitalWrite(C, LOW);
```

```
            digitalWrite(D, LOW);
            digitalWrite(E, LOW);
            digitalWrite(F, LOW);
          }
        }

        void loop()
        {
          timer.run();

          tegn(time2, HIGH);
          ciffer(10);

          tegn(time1, LOW);
          ciffer(11);

          tegn(minut2, HIGH);
          ciffer(12);

          tegn(minut1, HIGH);
          ciffer(13);
        }
```

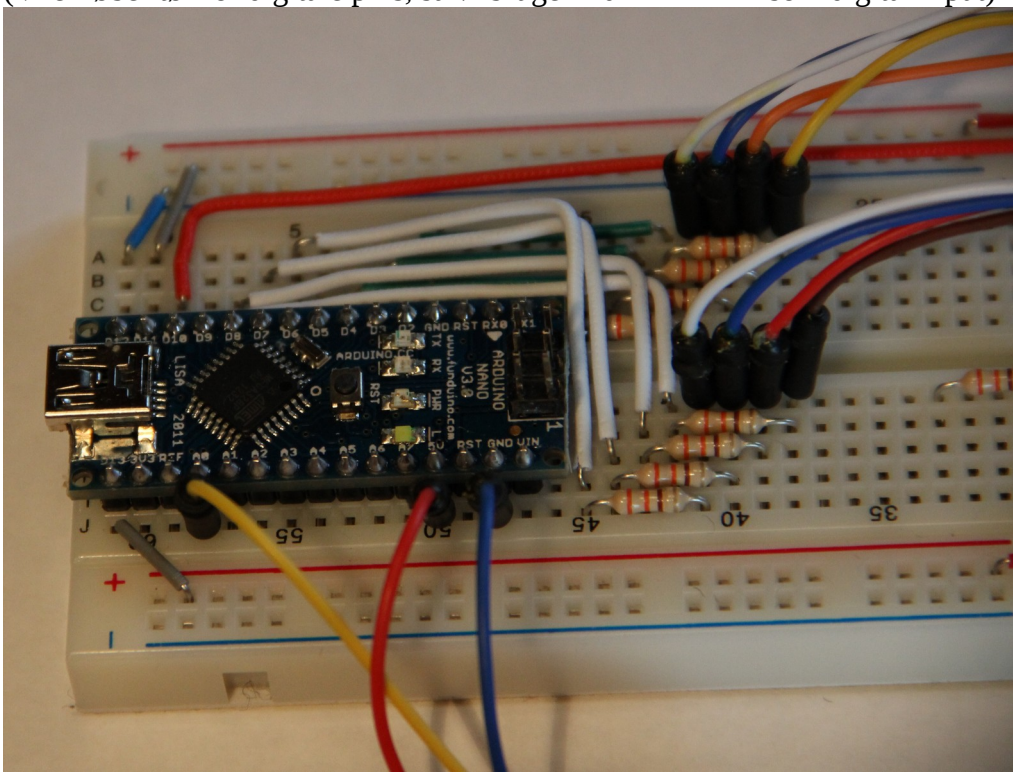
Tilføj en knap så man kan stille tiden.



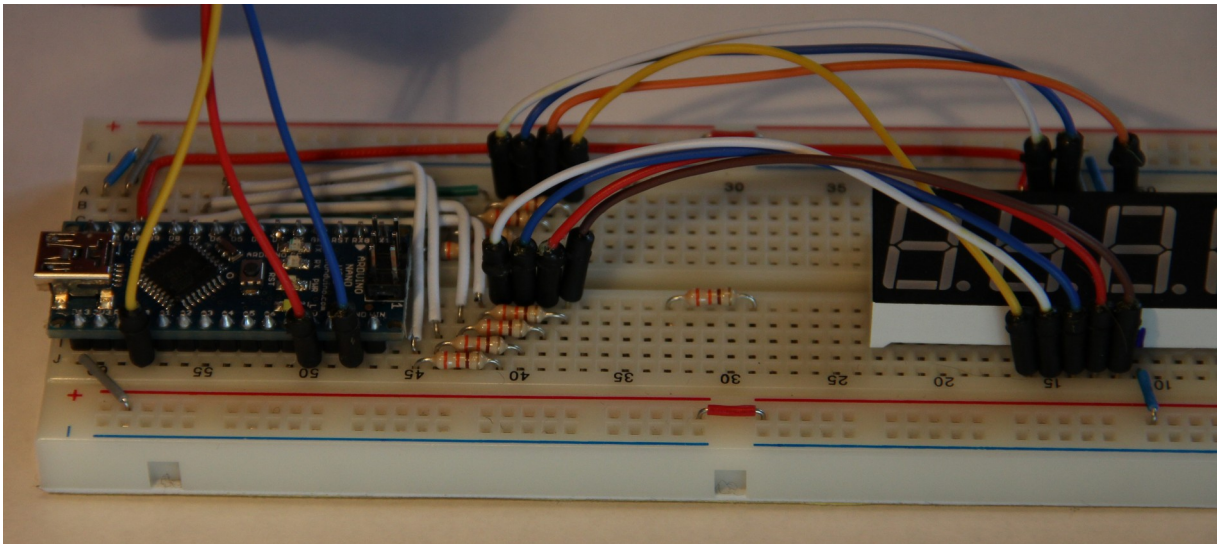
Forbind:

A0 -> Ben 1 på kontakten
5V -> Modstand -> Ben 2 på kontakten
GND -> Ben 3 på kontakten

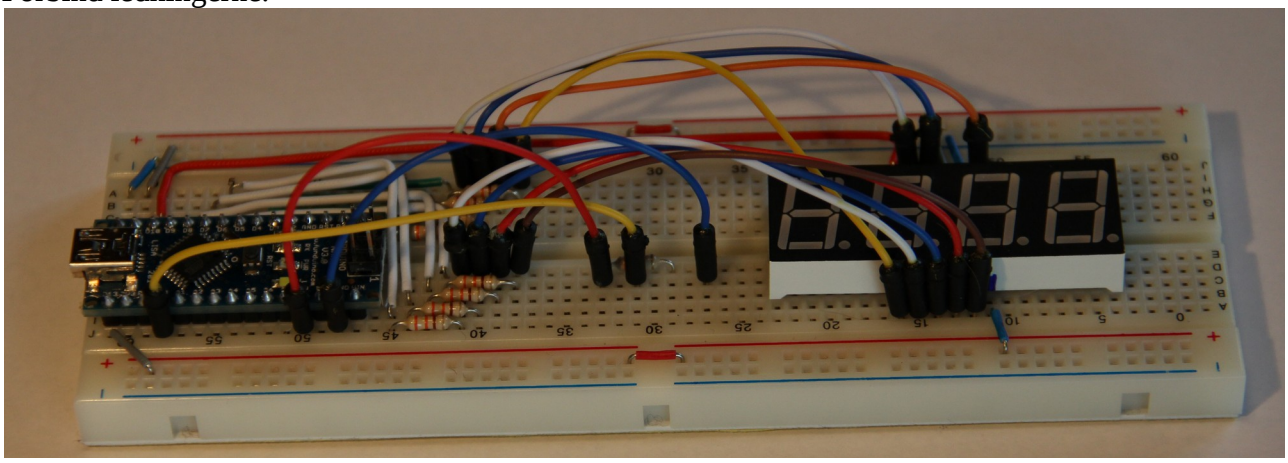
(Vi er løber tør for digitale pins, så vi bruger A0 == PIN 14 som digital input)



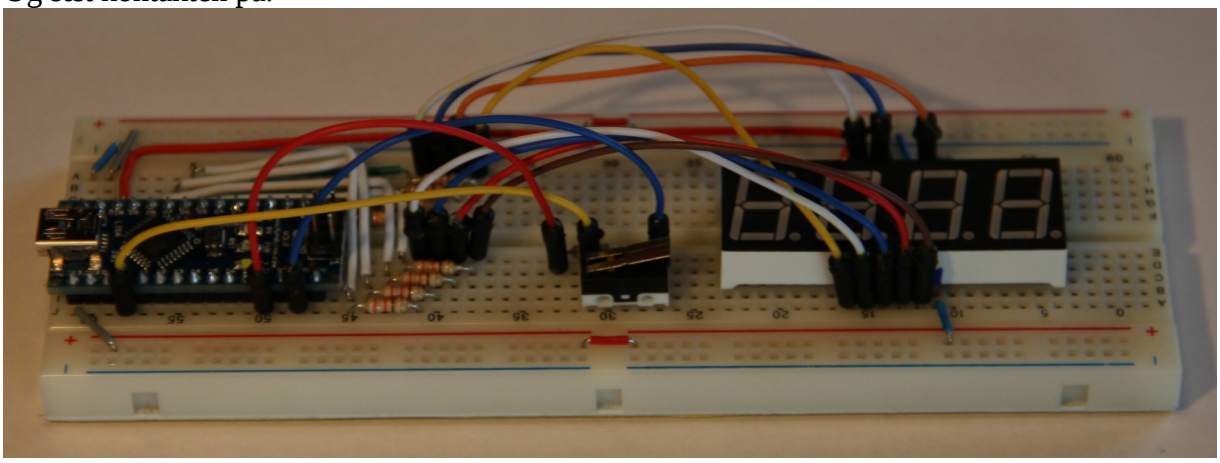
Placer modstanden et sted hvor der er plads



Forbind ledningerne.



Og sæt kontakten på.



Så skal Loop() bare udvides til at bruge knappen:

```
#include <SimpleTimer.h>
SimpleTimer timer;

// Lav variabler for tiden og sæt klokken til 11:58.
// 'Tid' tæller minutter siden midnat
int tid=60*11+58;
int time2, time1, minut2, minut1;
int timer_id;
int antal=0;

void setup()
{
  // Sæt PIN D2-D13 til at være OUTPUT
  for(int x=2; x<=13; x++) pinMode(x,  OUTPUT);

  // Setup button
  pinMode(14, INPUT);

  // Kald funktionen "pendul" hvert minut (60000 millisekunder)
  timer_id = timer.setInterval(60000, pendul);
  pendul();
}

void pendul()
{
  tid++;

  // Der er 1440 minutter på en dag
  if(tid >= 1440)
  {
    tid = tid - 1440;
  }
  time2 = tid/600;
  time1 = tid/60 - 10*time2;
  minut2 = tid/10 - 6*(10*time2 + time1);
  minut1 = tid - 10*minut2 - 60*(10*time2 + time1);
}

void ciffer(int D)
{
  // Tænd Ciffer D
  digitalWrite(D, HIGH);
  delay(1);
}

void tegn(int tal, int punktum)
{
```

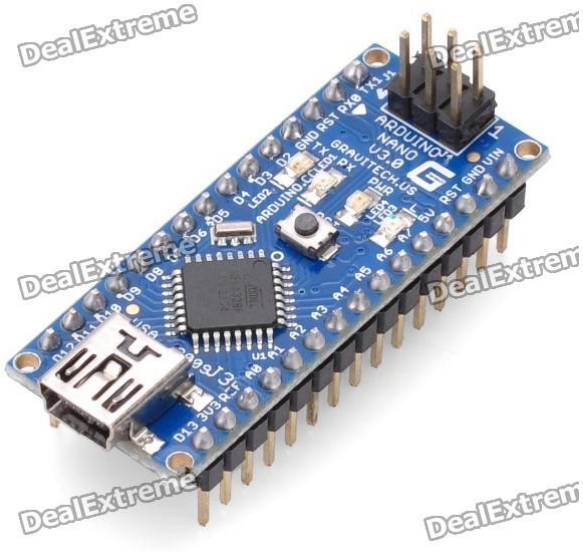
```
// Navngiv de PINs der svarer til segmenter i databladet
int E=2, D=6, DP=7, C=8, G=9, B=3, F=4, A=5;

// Sluk alle fire cifre
for(int x=10; x<=13; x++)  digitalWrite(x,  LOW);

// Sluk alle streger i display'et
for(int x=2; x<=9; x++) digitalWrite(x,  HIGH);

// Tænd eventuelt punktummet
digitalWrite(DP, punktum);

if(tal == 1)
{
  digitalWrite(B, LOW);
  digitalWrite(C, LOW);
}

if(tal == 2)
{
  digitalWrite(A, LOW);
  digitalWrite(B, LOW);
  digitalWrite(D, LOW);
  digitalWrite(E, LOW);
  digitalWrite(G, LOW);
}

if(tal == 3)
{
  digitalWrite(A, LOW);
  digitalWrite(B, LOW);
  digitalWrite(C, LOW);
  digitalWrite(D, LOW);
  digitalWrite(G, LOW);
}

if(tal == 4)
{
  digitalWrite(B, LOW);
  digitalWrite(C, LOW);
  digitalWrite(F, LOW);
  digitalWrite(G, LOW);
}

if(tal == 5)
{
  digitalWrite(A, LOW);
  digitalWrite(C, LOW);
  digitalWrite(D, LOW);
```

```
  digitalWrite(F, LOW);
  digitalWrite(G, LOW);
}

if(tal == 6)
{
  digitalWrite(A, LOW);
  digitalWrite(C, LOW);
  digitalWrite(D, LOW);
  digitalWrite(E, LOW);
  digitalWrite(F, LOW);
  digitalWrite(G, LOW);
}

if(tal == 7)
{
  digitalWrite(A, LOW);
  digitalWrite(B, LOW);
  digitalWrite(C, LOW);
}

if(tal == 8)
{
  digitalWrite(A, LOW);
  digitalWrite(B, LOW);
  digitalWrite(C, LOW);
  digitalWrite(D, LOW);
  digitalWrite(E, LOW);
  digitalWrite(F, LOW);
  digitalWrite(G, LOW);
}

if(tal == 9)
{
  digitalWrite(A, LOW);
  digitalWrite(B, LOW);
  digitalWrite(C, LOW);
  digitalWrite(D, LOW);
  digitalWrite(F, LOW);
  digitalWrite(G, LOW);
}

if(tal == 0)
{
  digitalWrite(A, LOW);
  digitalWrite(B, LOW);
  digitalWrite(C, LOW);
  digitalWrite(D, LOW);
  digitalWrite(E, LOW);
```

```
    digitalWrite(F, LOW);
  }
}

void loop()
{
  timer.run();

  tegn(time2, HIGH);
  ciffer(10);

  tegn(time1, LOW);
  ciffer(11);

  tegn(minut2, HIGH);
  ciffer(12);

  tegn(minut1, HIGH);
  ciffer(13);

  if (digitalRead(14))
  {
    antal++;
    delay(1);
  } else {
    antal=0;
  }
  if (antal == 1 || antal > 300)
  {
    pendul();
    timer.restartTimer(timer_id);
  }
}
```

Digitalur med sekunder og knapper

Til den næste opstilling skal du bruge:
1 stk Arduino med usb kabel
1 stk display modul med 8 cifre
Fladkabel med 10-polet stik i den ene ende og 5 krog-stik i den anden.



* Tag strømmen fra Arduino og forbind de fem kabler:
Arduino <-------> Display (venstre side)

5V <---- Rød -----> Ben 1 (VCC)
GND <--- Sort -----> Ben 2 (GND)
D9 <---- Hvid -----> Ben 3 (CLK)
D8 <---- Gul -----> Ben 4 (DIO)
D7 <---- Hvid/Grøn -----> Ben 5 (STB0)
* Dobbelt check din opstilling - få meget gerne Henrik eller Richard til at checke den efter. Sæt derefter strøm på Arduino.

Vi starter med et simpelt program der aflæser knapperne, tænder en lysdiode ud for alle indtrykkede knapper, og viser talkoden for de indtrykkede knapper i display'et:
* Start et nyt Arduino program med indhold:

```
#include <TM1638.h>
TM1638 module(8, 9, 7);

void setup()
{
}

void loop()
{
  int knapper;
  knapper = module.getButtons();
  module.setDisplayToDecNumber(knapper, 0);
  module.setLEDs(knapper);
}
```

Check programmet med "V" og læg det over på Arduino med "->".

Forklaring:
Display modulet har en "TM1638" kontrol IC på bagsiden, der styrer display, lysdioder og aflæser knapper. Den har selv lidt hukommelse hvor der står hvad den skal vise på display'et, så man ikke hele tiden selv skal sætte display værdien.
Styring af display modulet sker ved at læse/skrive i hukommelsen på TM1638 IC'en via 3 af de 5 ledninger (de sidste to er til strøm).
Der findes et bibliotek med funktioner til at styre et TM1638 display.

De første to linier:

```
#include <TM1638.h>
TM1638 module(8, 9, 7);
```

betyder "jeg vil bruge TM1638 biblioteket"
og "mit TM1638 modul er forbindet til DIO=D8, CLK=D9 og STB0=D7".

De funktioner du får med "#include <TM1638.h>" er:

```
module.clearDisplay();                      Slet alt.
module.clearDisplayDigit(3, false);  Slet ciffer 3 og sluk punktum
module.getButtons();                  Læs hvilke knapper der trykket på
module.setDisplay(values);            Styr display segmenter direkte
module.setDisplayDigit(3, 6, true);  Skriv '3' i ciffer 6 og tænd punktum
module.setDisplayToBinNumber(42,0);      Vis '42' i binær og sluk punktummer
module.setDisplayToDecNumber(42,0);      Vis '42' i decimal og sluk punktummer
module.setDisplayToHexNumber(42,0);      Vis '42' i hexadecimal og sluk punktummer
module.setDisplayToSignedDecNumber(-42,0); Vis '-42' i decimal og sluk punktummer
module.setDisplayToError();           Vis 'ERROR' i display
module.setDisplayToString("Thorbjorn"); Vis 'Thorbjorn' i display
module.setLED(TM1638_COLOR_GREEN, 3);     Tænd lysdiode 3 med grøn farve
module.setLEDs(42);                   Tænd lysdioder svarende til '42' i binær
module.setupDisplay(true, 7);                Tænd alt og sæt lysstyrke til 7 (0-7)
```

Den detaljerede beskrivelse (på engelsk) er her:
http://code.google.com/p/tm1638-library/wiki/Reference

Få tiden til at gå

Ligesom i den første opstilling bruger vi "SimpleTimer" biblioteket til at lave et pendul. Denne gang tæller vi dog sekunder istedet for minutter.

```
#include <TM1638.h>
#include <SimpleTimer.h>
TM1638 module(8, 9, 7);
SimpleTimer timer;

// Sekunder siden midnat - sæt tiden til 11:59:30
long tid=11*3600L + 59*60 + 30;

void setup()
{
  timer.setInterval(1000, pendul);
}

void pendul()
{
  tid++;
  // Der er 86400 sekunder på et døgn
  if (tid >= 86400)
  {
    tid = tid - 86400;
  }

  module.setDisplayDigit((tid/36000)%3, 1, false);
  module.setDisplayDigit((tid/3600)%10, 2, true);
  module.setDisplayDigit((tid/600)%6,   3, false);
  module.setDisplayDigit((tid/60)%10,   4, true);
  module.setDisplayDigit((tid/10)%6,    5, false);
  module.setDisplayDigit((tid)%10,      6, false);
}

void loop()
{
   timer.run();
}
```

Brug knapperne til at stille uret

Den simple løsning er at aflæse knapperne i "loop" og tælle tid op der:

```
void loop()
{
  timer.run();
  int knapper;
  knapper = module.getButtons();
  module.setLEDs(knapper);
  if (knapper == 64) tid = tid + 1;
  if (knapper == 32) tid = tid + 10;
  if (knapper == 16) tid = tid + 60;
  if (knapper == 8)  tid = tid + 600;
  if (knapper == 4)  tid = tid + 3600;
  if (knapper == 2)  tid = tid + 36000;
}
```

Det virker bare ikke ordenligt - prøv en gang.
Der går to ting galt:
1) Når man trykker på en knap, når den ind i "loop" flere tusinde gange inden man får sluppet knappen igen, og tæller derfor op flere tusinde gange.
   Vi har brug for at vente i bunden af "loop" indtil knappen er sluppet igen. Det kan vi gøre med en while-løkke.

```
while(knapper > 0) knapper = module.getButtons();
```

   En while-løkke er en if-sætning der bliver gentaget i det uendelige, indtil test-betingelsen "knapper > 0" ikke længere er sand.

2) Display'et bliver ikke opdateret når man trykker på knappen, men først næste gang "pendul"'et svinger.
   Vi har brug for at lave samme opdatering som "pendul" også laver. For ikke at skrive det samme to gange deler vi "pendul" op i to funktioner - "opdater" der opdaterer display, og "pendul" der tæller 'tid' op og derefter kalder "opdater".
   Så kan vi nemlig også kalde "opdater" fra "loop".

Det endelige program ser sådan ud:

```
#include <TM1638.h>
#include <SimpleTimer.h>
TM1638 module(8, 9, 7);
SimpleTimer timer;

// Sekunder siden midnat - sæt tiden til 11:59:30
long tid=11*3600L + 59*60 + 30;
```

```
void setup()
{
  timer.setInterval(1000, pendul);
}

void pendul()
{
  tid++;
  updater();
}

void updater()
{
  // Der er 86400 sekunder på et døgn
  if (tid >= 86400)
  {
    tid = tid - 86400;
  }

  module.setDisplayDigit((tid/36000)%3, 1, false);
  module.setDisplayDigit((tid/3600)%10, 2, true);
  module.setDisplayDigit((tid/600)%6,   3, false);
  module.setDisplayDigit((tid/60)%10,   4, true);
  module.setDisplayDigit((tid/10)%6,    5, false);
  module.setDisplayDigit((tid)%10,      6, false);
}

void loop()
{
  timer.run();
  int knapper;
  knapper = module.getButtons();
  module.setLEDs(knapper);
  if (knapper == 64) tid = tid + 1;
  if (knapper == 32) tid = tid + 10;
  if (knapper == 16) tid = tid + 60;
  if (knapper == 8)  tid = tid + 600;
  if (knapper == 4)  tid = tid + 3600;
  if (knapper == 2)  tid = tid + 36000;
  if (knapper > 0) updater();
  while(knapper > 0) knapper = module.getButtons();
}
```