Arduino opgaver:

Samlede opgaver til teknologi og El-teknik

Hop til opgave: Find hjælp, Indledning, Kit Vers.1, Kit Vers.2, Blinkende lysdioder, Input fra knap, Intern Pullup, If – Else, Variabel blinkfrekvens, For-Loop, Binære tal, Random lysdioder, Fadning af lysdiode, LED styret af Potentiometer, RGB-Kit, Knight Rider, LED-Cube, Piezo Beeper, Morsekode, Stepmotor,

LCD-Display, Multi LCD, Debug-vindue, Data fra Debugvinduet til Ardu, Mål analog spænding, Vis analog spænding på LCD, LED styret af Potentiometer, Mål temperatur, Ur-Program, Keypad,

Termoprinter, Seriel transmission til "Hej Mor", 7-Segment, 7-segment-Multiplexing, Pernille-Display, Dot Matrix Display, RF-ID, Servomotor, Timer-interrupt, Stopur, Udvidelse af Pin-antal, Flere Inputs, Intern EEPROM, Ekstern EEPROM, Små Kits opgaver,

Kopiering af kode med farve til rapportskrivning i Word,

Find Hjælp:

For at få hjælp til at lave opgaverne, brug Arduino-kompendiet, eller søg på nettet!

På Nettet søges fx med "Arduino" + et søgeord mere
Se evt. YouTube. Fx på (rigtig god) https://www.jeremyblum.com/category/arduino-tutorials/

Top ↑

Indledning:

I dette dokument er der inspiration til en række programmerings-opgaver til Arduino. Opgaverne er fælles for 3. del El og teknologi. Dvs. der er nogle af opgaverne der er lette, og andre er ret komplekse. Men 2. del må gerne forsøge sig med dem alle !!

Opgaverne starter i den lette ende, og bliver i nogen grad sværere og sværere.

Opgaverne kan laves ved at opbygge et kredsløb på et fumlebrædt. – Eller ved at bruge én af Arduino-Kit-ene. Kittene har monteret en række komponenter, og gør det meget hurtigt at opbygge og teste forskellige programmer og forsøgsopstillinger.

Der findes flere versioner af kittene, med lidt forskellige layout.

På alle er der et LCD-display med 4 linjer á 20 karakter, 8 lysdioder, et Keypad, et potentiometer, nogle trykknapper mm.

På **version 2** er der kun 3 trykknapper, men til gengæld en LM35 temperatur-transducer og to lydgivere. Den ene giver en lyd på ca. 2 kHz blot den får 5 Volt, fx fra en outputpin på Arduinoen. Den anden skal have tilført den frekvens, der skal høres.

For hjælp til at forbinde Kittene korrekt til Arduinoerne, se dokument med printudlæg over kittene.

http://vthoroe.dk/Elektronik/Instrumenter/Arduino_kits.pdf

Vælg blandt opgaverne, lav dem, skriv kommentarer i kildeteksten, dokumenter med flowchart og aflever !!

Der er en række opgaver. Vælg – og kom så langt, du vil / kan !!!

Start med de første opgaver, - og gå så fremad!

Mange af opgaverne kan løses ved at starte med et færdigt eksempel. Enten en af de medfølgende i IDE-en, eller noget fundet på nettet. Brug så eksemplerne som inspiration, og prøv at modificere programmerne.

Til de fleste opgaver er der et eksempel, der kan C&P ind i Arduino IDE'en og afprøves. Men ideen er jo, at man selv skal bearbejde, ændre i programmet eller tilføje mere funktionalitet.

Top ↑

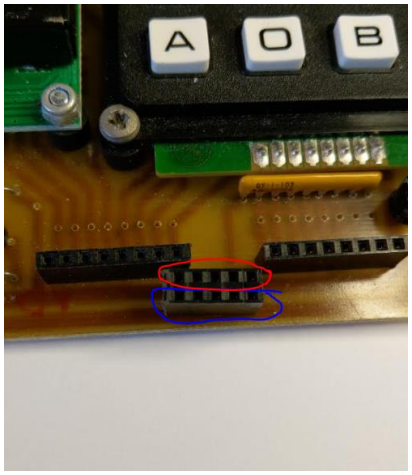
Bemærk, at der findes flere versioner af Arduinokittene !!

Seneste ændring pr. Januar 2021:

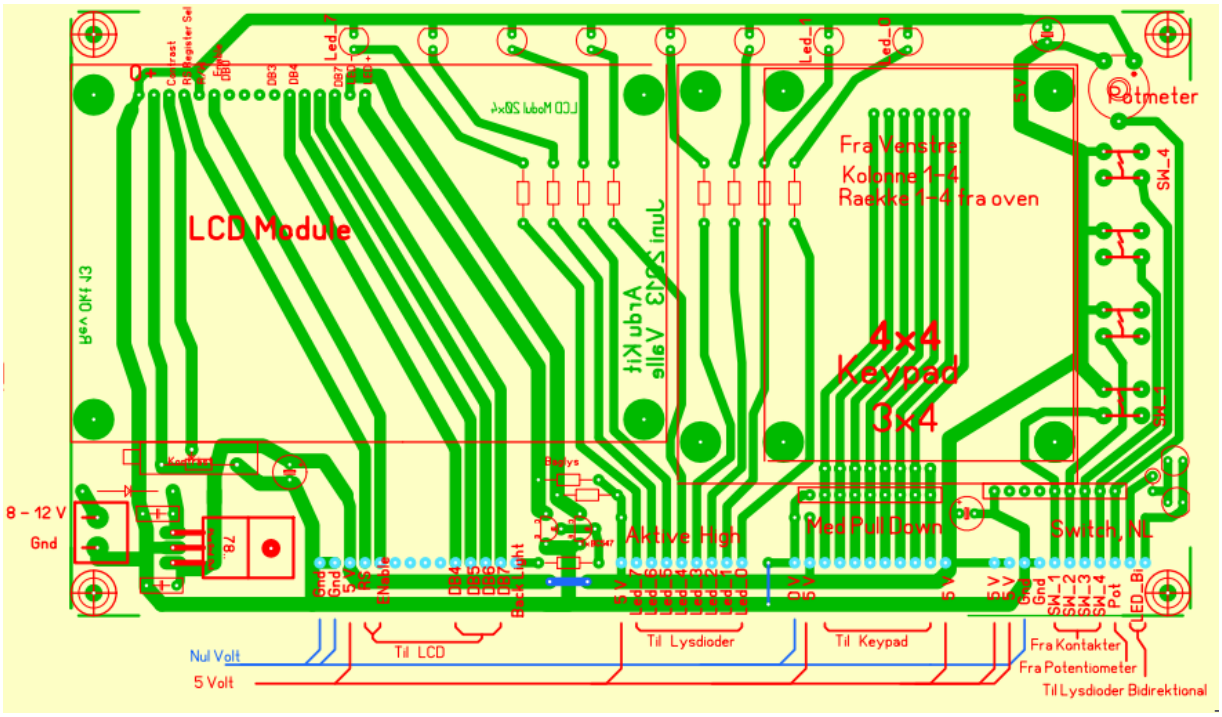
Det har mange gange været et problem at finde ud af at forbinde +5Volt og Gnd til kittene.

Ligeledes har vi mange gange manglet flere +5Volt pins.

Derfor er der nu tilføjet nogle ekstra +5 Volt-pins. De er markeret med rød på billedet.
Og ligeledes flere Gnd markeret med blå.

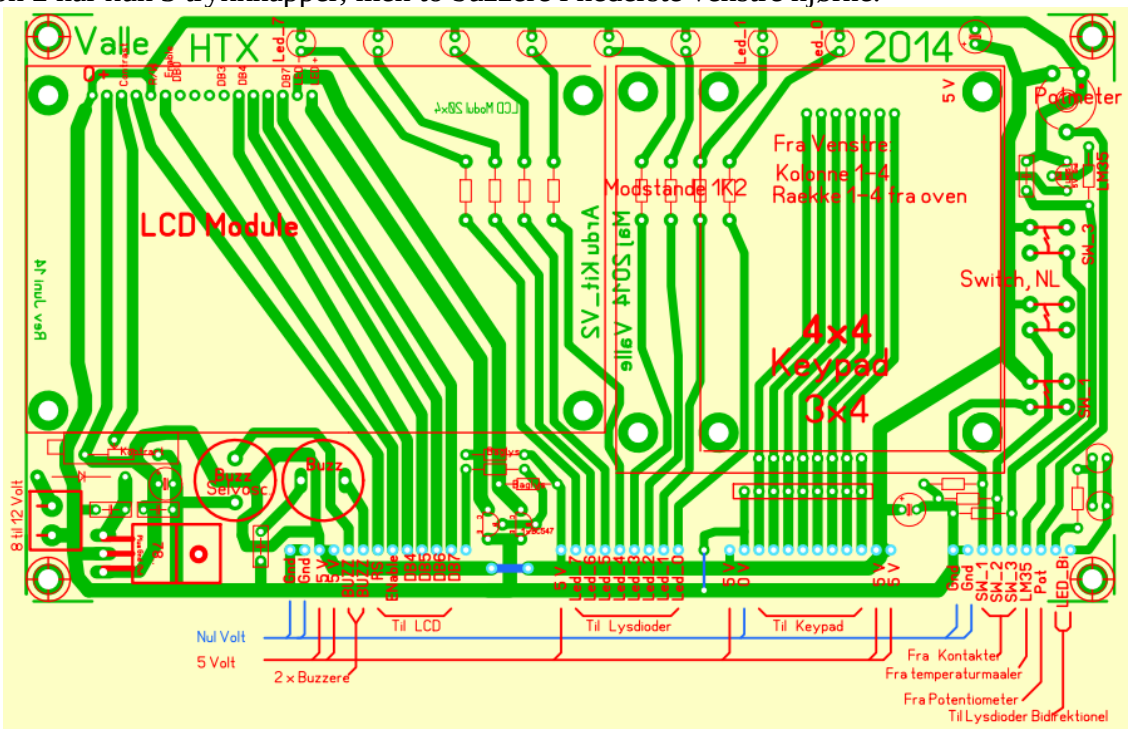


Kit Version 1:



Kit Version 2:

Version 2 har kun 3 trykknapper, men to buzzere i nederste venstre hjørne.



På Version 2 er der tilføjet 2 Buzzere under LCD-en. Den ene giver lyd på ca. 2 KHz blot der tilsluttes 5 Volt. Den anden skal have tilført en frekvens svarende til den frekvens, man vil høre.

Top ↑

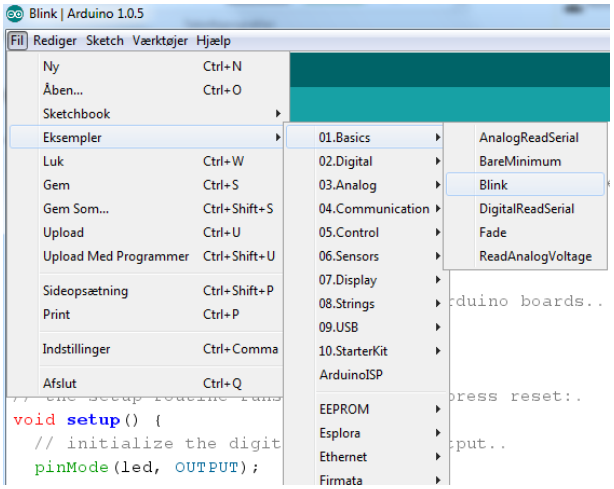
Blinkende Lysdioder

Hent sketchen ”Blink”.
Fil\Eksempler\Basics\Blink.

Programmet blinker en lille lysdiode på Arduino-bordet koblet til pin 13.

Lav lidt om på blink-intervallet.

Omskriv fx programmet, så der kommer 2 blink, efterfulgt af en pause.



Få lysdioden til at blinke morse-tegnene ” FS ” som man også kan høre fra tågehornet i fyret på Kalk-grund sydøst for Sønderborg.

Brug et kit, eller monter 8 LED på et fumlebrædt

Husk også at forbinde 0 Volt. Det er mærket Gnd for Ground.

Brug igen blinkprogrammet men udvid det til at få alle 8 LED til at lyse. Fx skiftevis med 1 sekund imellem hver.

Brug fx fra pin 6 og op til pin 13. Obs: brug ikke pin 0 og 1, da de bruges til at kommunikere via USB-kablet.

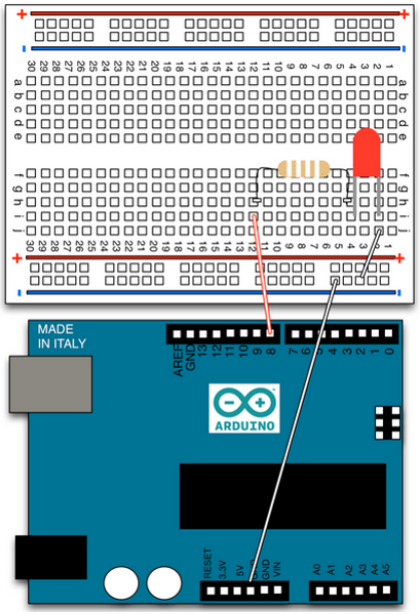
Husk at definere alle pins som outputs i setup!

Prøv at lave om på blinkfrekvensen.

Her er vist et par eksempler på, hvordan man kan forbinde eksterne lysdioder på et fumlebrædt.

Her dog kun vist med 1 LED.

Husk formodstand.



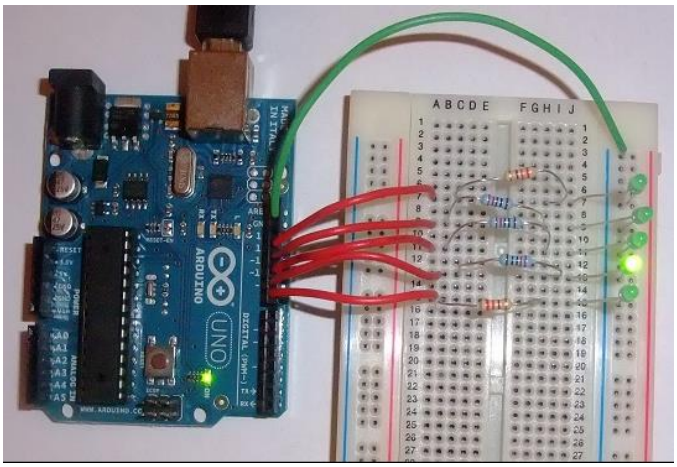
Her et Fumlebrædt set forfra.

Forbindelserne under hullerne er mærket op med rødt.

For mere se fx: http://vthoroe.dk/Elektronik/Instrumenter/Fumlebraedt.pdf

Her er vist et eksempel med flere lysdioder.

Husk at forbinde Gnd!! 0 Volt,



Lad delayet være bestemt af en variabel, og indret programmet så blinkfrekvensen ændres for hver gennemløb.

I loop-sløjfen tilføjes fx følgende:

```
delay(time);
      time = time - 10;
      if(time < 5) time = 5;
```

Indret nu programmet, så variablen delay varierer op og ned mellem 10 og 1000 (mS).

Monter 2 knapper på hver sin inputpin. Brug evt. testkittet. Den har pull-down-modstande på trykknapperne til højre.

Den ene knap skal kunne skrue op, og den anden knap ned for blink-frekvensen på en lysdiode.

Find evt. hjælp i eksemplet: Fil\Eksempler\Digital\Button. Eller senere her i dokumentet !!

Eller brug

```
if(digitalRead(input_1) == 1)      // 2 lighedstegn efter hinanden betyder
                                   //  "Logisk lig med"
   {
       time = time - 10;
       if(time < 5) time = 5;      //  time kan ikke blive mindre end 5
   }
```

Evt. kan man læse en spænding fra potentiometeret, og lade delayet være styret af den læste værdi. Herved kan man ændre blinkfrekvensen ved at dreje på potentiometeret.

Top ↑

Input fra knap

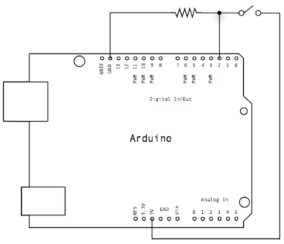
Lav et program, der får en lysdiode til at lyse, så længe, der er trykket på en knap:

Her er der behov for, at programmet kan læse en kontakt, forbundet til en input-pin.

Her et eksempel på forbindelser.

Bemærk, at der er trykknapper på mine kits.

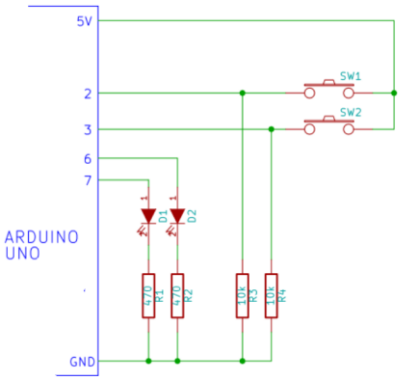
Kittet skal have 5 Volt, 0 Volt, og en forbindelse fra en pushbutton. På billedet til højre er vist hvordan et kredsløb kan se ud.



Når der trykkes på knappen, bliver signalet på pin 2 højt.

Når der trykkes på en knap, kan der løbe strøm ned gennem modstanden. Herved bliver spændingen over modstanden – og dermed også på pin-en høj. (5 Volt).

Hvis modstanden undlades, vil ledningen hen til pin 2 og 3 svæve når der ikke trykkes på knappen. Dvs. den ikke er forbundet til noget andet end en stump ledning. Dvs. der meget let dannes 50 Hz signal i ledningen. Det sker på grund af elektriske og magnetiske felter fra vores elnet.

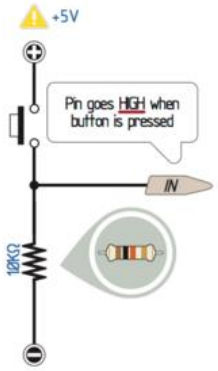


Knapper, - eller buttons – eller switch-es fås i forskellige udformninger. Til forskellige formål.

Forbindelser mellem Unoen og mine kits kan laves sådan!

Nu skal der bruges en program-funktion der kan læse en pin. Dvs. læse om den er lav eller høj:

Program-Eksempler:




```
// definer input pin

int buttonpin=2; // Placeres før Setup!!

// I setup defineres pin som input!

pinMode(buttonpin, INPUT);


// i Loop placeres kode, der skal gentages:

if (digitalRead(buttonpin) == HIGH) {

    // do something

}
```

Ovenfor er der brugt en ” if ” – struktur!!

En ”if-algoritme” får programmet til at teste om en betingelse er opfyldt. Og hvis den er, udføres nogle programlinjer mellem { og }.

Her et andet eksempel:

```
void loop()
{
  val = digitalRead(inPin);   // read the input pin

  digitalWrite(ledPin, val);    // sets the LED to the button's value

if (val == 1) {  // der skal to lighedstegn til for at der sammenlignes.

// Do something
    }
}
```

Find evt. koden her: https://www.arduino.cc/en/tutorial/pushbutton

Et eksempel mere:

Her er der brugt en While-struktur i stedet for en if:

Se eksempler på kodestrukturer eller algoritmer på: http://vthoroe.dk/Arduino/Algoritmer.pdf

Enable Intern pullup:

For at der kan kobles en switch på en indgang, skal der normalt bruges en extern pull up, - eller pull down-modstand. Men i Arduino-uC'en er der mulighed for at enable intern pull up-modstand på ca. 20 Kohm.

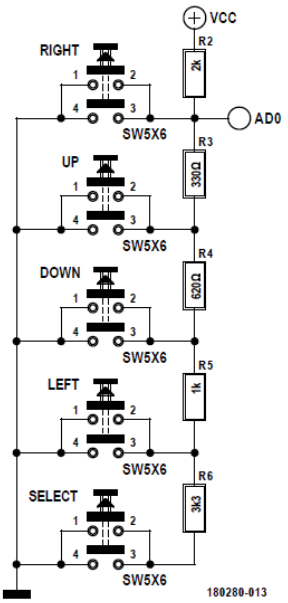
Definer først en pin som input, og gør den dernæst høj. (digitalWrite(inputpin, HIGH))

Se kompendiet Tips og Trix http://vthoroe.dk/Elektronik/Arduino/Tips_og%20Trix.pdf

Eksempel:

```
void loop()
{

  while( digitalRead(5) == 1 )          // while the button is pressed
    {
    //blink
    digitalWrite(3,HIGH);
    delay(1000);
    digitalWrite(3,LOW);
    delay(1000);
    }
}
```

// Bonus: Brug en analog input til at vælge blandt flere knapper:




```
// Læs analog pin

void get_key_pushed()
{
  X = analogRead(A0) / 10; // values from 0 to 102
  delay(100); // key debounce

  if (X < 5) X = 2;           // Right
  if (X > 5 && X < 20 ) X = 3; // Up
  if (X > 20 && X < 35 ) X = 4; // Down
  if (X > 35 && X < 55 ) X = 1; // Left
  if (X > 55 && X < 85 ) X = 5; // Select
}
```

//Kilde: https://it.emcelettronica.com/telemetria-in-half-duplex-a-2-4-ghz

Se også fx her:

Top ↑

Brug af – If – Else Eksempel:

```
//  If - Eksempel:

if (x > 120){
  digitalWrite(LEDpin1, HIGH);
  digitalWrite(LEDpin2, HIGH);
}


//  If else

if (x < 500)
{
  // action A
}
Else       // Else-delen kan udelades
{
  // action B
}
```

Og så hele koden:

```
/* Basic Digital Read, Kodeeksempel:
```

```
 * ------------------
 *
 * turns on and off a light emitting diode (LED) connected to digital
 * pin 13, when pressing a pushbutton attached to pin 7. It illustrates the
 * concept of Active-Low, which consists in connecting buttons using a
 * 1K to 10K pull-up resistor.
 *
 * Created 1 December 2005
 */

int ledPin = 13; // choose the pin for the LED
int inPin = 7;   // choose the input pin (for a pushbutton)
int val = 5;     // variable for reading the pin status. Start value = 5

void setup() {
  pinMode(ledPin, OUTPUT);  // declare LED as output
  pinMode(inPin, INPUT);    // declare pushbutton as input
}

void loop(){
  val = digitalRead(inPin);  // read input value
  if (val == HIGH) {         // check if the input is HIGH (button released)
    digitalWrite(ledPin, LOW);  // turn LED OFF
  } else {
    digitalWrite(ledPin, HIGH);  // turn LED ON
  }
}
```

// Bemærk: if (val == HIGH) { }

Hvis der kun bruges 1 lighedstegn, får "val" tildelt værdien HIGH, - som er lig 1.

Men bruges 2 lighedstegn, udføres en test, og koden mellem { } udføres kun hvis udfaldet af testen er sandt.

Afprøv ovenstående program!!

Udvid ovenstående så der er to lysdioder, der lyser på skift, dvs. den ene, hvis der ikke trykkes, og den anden hvis der trykkes.

Lav et program, så et kort tryk på en knap får en lysdiode til at lyse i 5 sekunder.

Top ↑

Variabel Blinkfrekvens

Lav et program, hvor man ved hjælp af 2 knapper kan lave variabel pauselængde mellem blink i en lysdiode.

Ideen er nu, at den ene knap skal kunne skrue op, og den anden knap ned for blink-frekvensen på en lysdiode.

Monter 2 knapper på hver sin input-pin. Fx på pin 4 og 3.

Husk modstande til knapperne, og formodstande for Lysdioder hvis de monteres på et fumlebrædt.

Husk også at definere pins som input.

Strategi: Lad pauselængden delay() være defineret i en variabel

Variabelnavnet kan så bruges i programmet i stedet for en fast værdi for en pause.

```
/*   Kodeeksempel:

Programbeskrivelse:


*/

// Def af variable til at holde et pinnummer.

const byte upPin = 4;      // the number of the pushbutton pin
const byte downPin = 3;      //

//  Vi skal også bruge en variabel til at indeholde en værdi, der skal bruges
i delay-funktionen!

int delayvalue = 100;   // Startværdien er 100. en Integer kan højest være
                                //65.535

byte buttonState = 0;    // skal bruges til at læse værdien af en knap,
                                om den er lav eller høj.


void setup() {

  pinMode(upPin, INPUT);    // initialize the button pin as an input:
  pinMode(upPin, INPUT);    // initialize the button pin as an input:

  //  Og alle LED-Outputpins skal jo selvfølgelig være output.

}
```

```
  //  **********************

void loop() {

  digitalWrite(13, HIGH );
  delay(delayvalue);
  digitalWrite(13, LOW );
  delay(delayvalue);

  buttonState = (digitalRead(upPin));

  if (buttonState == HIGH) {

    delayvalue++; // adder 1 til værdien
                     ( det same som delayvalue = delayvalue + 1 )

  }

  buttonState = (digitalRead(downPin));
  if (buttonState == HIGH) {

    delayvalue--; // træk 1 fra værdien

  }
}
```

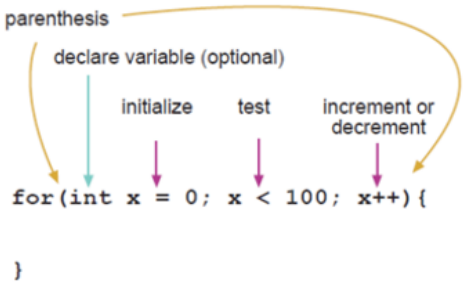
Undersøg programmet – og test det, og prøv at ændre i det.

Top ↑

For-loop:

En for-loop bruges til at udføre noget et antal gange.

En ” For ” løkke er en struktur, der gentages et antal gange. Den skal tolkes på følgende måde:



Variablen x starter her med at få tildelt værdien 0, og koden mellem { og } udføres første gang.

Herefter testes om x i dette tilfælde er < 100.

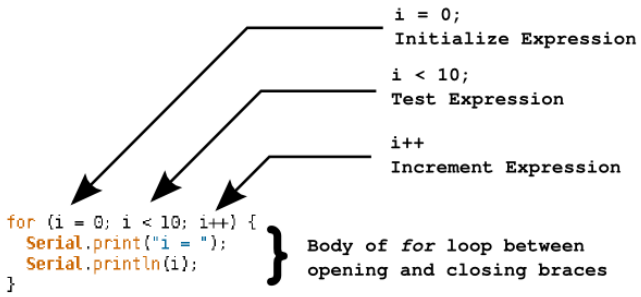
Hvis sand øges x med 1.

x++ er det samme som at skrive: x = x+1.

Herefter gentages koden mellem { og }.

Værdien af variablen x kan så også bruges i koden mellem { og }.

Eksempel:

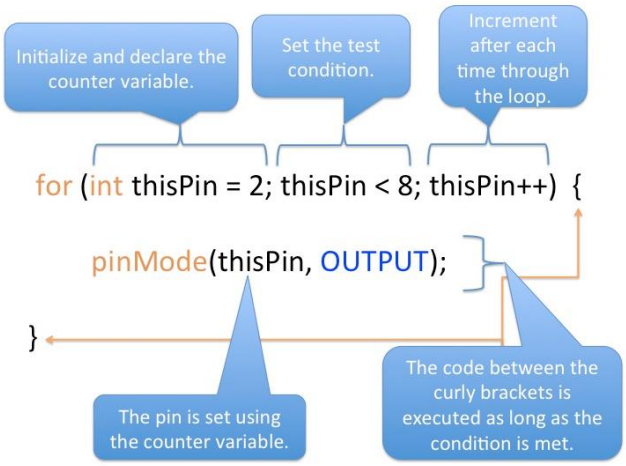


Her mangler en anden forklaring !!

Brug ovenstående til at få lysdioder koblet til pin 2 til 9 til at blinke på skift.

```
For ( i = 2; i <= 9; i++ {
    digitalWrite( i, HIGH);
    delay(100);
    digitalWrite( i, LOW);
    delay(100);
 }
```

Her et andet eksempel på en for-loop. Den bruges her til at gøre pin 2 til pin 7 til outputs.



Her bruges en for-loop til først at definere pins som outputs, - og dernæst til at tænde tilsluttede lysdioder i en løkke-struktur!!

```
/*
For Loop
```

```
 Demonstrates the use of a for() loop.
 Lights multiple LEDs in sequence, then in reverse.

 The circuit:
 * LEDs from pins 2 through 7 to ground

 created 2006  by David A. Mellis
 modified 30 Aug 2011  by Tom Igoe

This example code is in the public domain.

 http://www.arduino.cc/en/Tutorial/ForLoop
 */

int timer = 100;           // The higher the number, the slower the timing.

void setup() {
  // use a for loop to initialize each pin as an output:
  for (int thisPin = 2; thisPin < 8; thisPin++) {
    pinMode(thisPin, OUTPUT);
  }
}

void loop() {
  // loop from the lowest pin to the highest:
  for (int thisPin = 2; thisPin < 8; thisPin++) {
    // turn the pin on:
    digitalWrite(thisPin, HIGH);
    delay(timer);
    // turn the pin off:
    digitalWrite(thisPin, LOW);
  }

  // loop from the highest pin to the lowest:
  for (int thisPin = 7; thisPin >= 2; thisPin--) {
    // turn the pin on:
    digitalWrite(thisPin, HIGH);
    delay(timer);
    // turn the pin off:
    digitalWrite(thisPin, LOW);
  }
}
```

Her et eksempel på et array brugt til at definere pins.

```
// placeres før setup()

int pinArray[] = {2, 3, 4, 5, 8, 9};  // definer array-element # 0 til 5 !!

// placeres i setup()


  for (count=0;count<6;count++) {
    pinMode(pinArray[count], OUTPUT);
  }
```

```
// placeres i loop()

  for (count=0;count<6;count++) {
   digitalWrite(pinArray[count], HIGH);
   delay(100);
   digitalWrite(pinArray[count], LOW);
   delay(100);
  }
```

Top ↑

Binære tal

Her skal der defineres en variabel, fx en integer, og dens værdi tælles op til 255 i en løkke. Brug fx:

```
 for (int i=0; i<256; i++) {
     // Do
 }
```

Tallets binære værdi skal vises på de 8 lysdioder.

```
/*
 Binære tal på 8 pins
 @ Valle
 28/01-14
 */

// Def af pins
int led7 = 13;
int led6 = 12;
int led5 = 11;
int led4 = 10;
int led3 = 9;
int led2 = 8;
int led1 = 7;
int led0 = 6;

int time = 100;    // definer altid alle værdier her

void setup()
{
    pinMode(led0, OUTPUT);     // initialize the digital pin as an output.
    pinMode(led1, OUTPUT);
    pinMode(led2, OUTPUT);
    pinMode(led3, OUTPUT);
```

```
    pinMode(led4, OUTPUT);
    pinMode(led5, OUTPUT);
    pinMode(led6, OUTPUT);
    pinMode(led7, OUTPUT);
}

void loop()
{
    for (int i=0; i<256; i++)
    {
        digitalWrite(led0, bitRead(i, 0)); // læs bit 0 i variablen i
        digitalWrite(led1, bitRead(i, 1));
        digitalWrite(led2, bitRead(i, 2));
        digitalWrite(led3, bitRead(i, 3));
        digitalWrite(led4, bitRead(i, 4));
        digitalWrite(led5, bitRead(i, 5));
        digitalWrite(led6, bitRead(i, 6));
        digitalWrite(led7, bitRead(i, 7));
        delay(time);
    }
}
```

Lav koden om, så pinsene defineres vha. et array !!

Udvid nu programmet, så tallet i sendes til PC-ens debugvindue.

```
Serial.begin(9600);

 Serial.println(i);
 Serial.print(i.BIN);

 // Tæl både op og ned !

 //Forlæg kode til en subrutine

     for (int j=0; j<256; j++)
    {
        update_leds(j);
    }

 // Subs
void update_leds(int i)
{

}
```

Alle 8 bit:

Studer og brug følgende program.

Der er bare et problem, der skal håndteres, idet pin 0 og pin 1 bruges til kommunikation mellem PC-en og Arduinoboardet. Derfor skal programmet laves om, så der fx bruges fra pin 2 og opad.

Og der mangler også et passende delay.

```
// Digital 0 to7 set as outputs, then on/off using digitalWrite()

void setup()
{
  for (int a=2; a<10; a++)
  {
    pinMode(a, OUTPUT);
  }
}

void loop()
{
  for (int a=2; a<10; a++)
  {
    digitalWrite(a, HIGH);
  }
  for (int a=2; a<10; a++)
  {
    digitalWrite(a, LOW);
  }
}
```

Top ↑

Få lysdioder til at lyse Random

```
void loop(){
  int i=random(4);
  digitalWrite(ledPin[i], HIGH);
  delay(100);
  digitalWrite(ledPin[i], LOW);
  delay(100);
}
```

Mangler lidt forklaring, men

Se http://vthoroe.dk/Arduino/Algoritmer.pdf

Undersøg ”random”-funktionen. Få programmet til at køre. Og udvid derefter til 6 LED.

Top ↑

Fadning af lysdioder

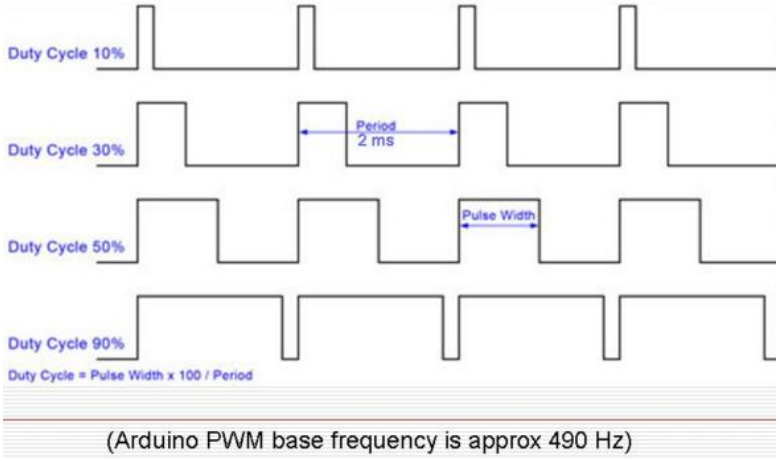
I Arduino-verdenen er der indbygget mulighed for at lave en slags pseudo-dæmpning af lysdioder. Det foregår ved at pulse på en udgang med en fast frekvens, men så få ” ON ” – tiden til at variere.

Begrebet kaldes ”PulsBredde-Modulation”, forkortet PWM.

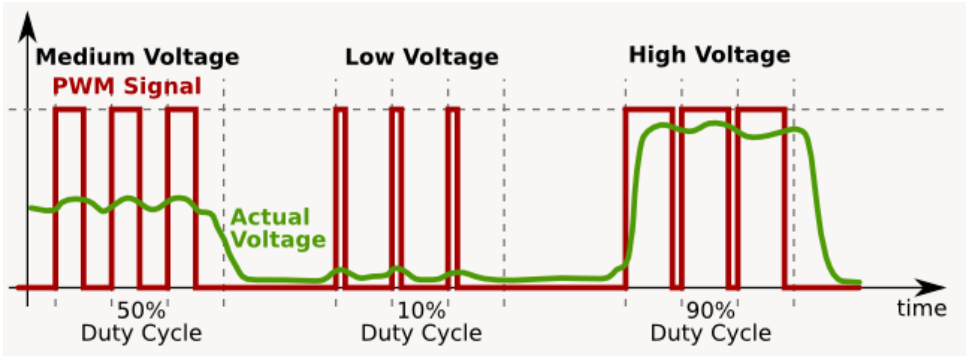
Og begrebet ”Duty Cycle” beskriver i hvor mange procent af en periodetid, et signal er høj.

Det kan bruges til at dæmpe (= fade) lyset i en lysdiode.

Det illustreres på denne graf:



Den grønne graf illustrerer lysstyrken i en lysdiode, som den opleves ved pulsbredde-modulation.



I Arduino – verdenen er denne mulighed for at bruge PWM indbygget i boardet. Men kun på de output, der er mærket med en bølgelinje ~.

Funktionen virker på et Uno-board på pin 3, 5, 6, 9, 10 og 11.

PWM-outputtene pulses automatisk med ca. 400 Hz, og dutycyclen styres ved at sende en værdi mellem 0 og 255 til en pin.

Men man skal i stedet for digitalWrite bruge:

```
analogWrite(pin, value); // Value kan have værdien fra 0 til 255.
```

Hvis en lysdiode på kittet forbindes direkte til PWM-outputtene, kan man nu dæmpe, eller fade lyset.

```
/*
 Fadning af dioder
 af: Valle
 d. 28/01-14
  */

// Def af pins

int led1 = 10;
int led2 = 11;

int time = 100;
int delaytime = 10;

void setup()
{
    pinMode(led1, OUTPUT);
}

void loop()
{
    analogWrite(led1, time);

    time++;
    if (time > 250) time = 5;
    delay(delaytime);
}
```

```
// Ekstra:

 analogWrite(led2, 255-time);

 while(time<250) {
 analogWrite(led1, time);
 analogWrite(led2, 255-time);
 time++;
 delay(delaytime);
 }
 while (time > 5) {
 analogWrite(led1, time);
```

```
analogWrite(led2, 255-time);
time--;
delay(delaytime);
}

// Måske kan man lade lysstyrken være afhængig af et potmeter

val = analogRead(analogPin);  // læs input pin, fx fra pin A0
analogWrite(ledPin, val / 4); // analogRead, værdier kan være fra 0 til 1023
                              //analogWrite værdier fra 0 til 255
```

Først skal det virke med 1 lysdiode, - dernæst skal den ene lysdiode øge sin lysstyrke mens den anden svækkes.

Lav derefter programmet om, så det er LED'ens lysstyrke, der ændres ved at trykke på nogle knapper. Brug PWM-output.

Lysdiode styret af potentiometer

Monter et eksternt potentiometer til en analog indgang. Den skal have 5 Volt og nul, og midterbenet skal fx til indgang A0.

Eller brug kittet med et potentiometer.

Lad den læste værdi afgøre hvor hurtigt en lysdiode blinker. – og eller hvordan den fades.

Obs: De værdier, der læses fra en analog indgang med en spænding mellem 0 og 5 Volt bliver til et tal mellem 0 og 1023. Dvs. der læses 10 bit. Men de værdier, der kan skrives til en PWM-udgang er kun på 8 bit.

Her er eksempler på, hvad der kan bruges af kode:

```
const int analogIndgang = A0;   //Definer indgangnummer

int analogVaerdi = 0;           // definer en variable, giv den værdien 0
analogVaerdi = analogRead(analogIndgang);  // læs værdi til variabel
analogVaerdi = analogVaerdi / 4 // omregn til max 8 bit.
```

analogVaerdi kan nu bruges i et program til fx at bestemme blinkfrekvens – eller fade-value.

Top ↑

RGB-Kit:

Et RGB-kit kan bruges til at eksperimentere med Røde, Grønne og Blå lysdioder.

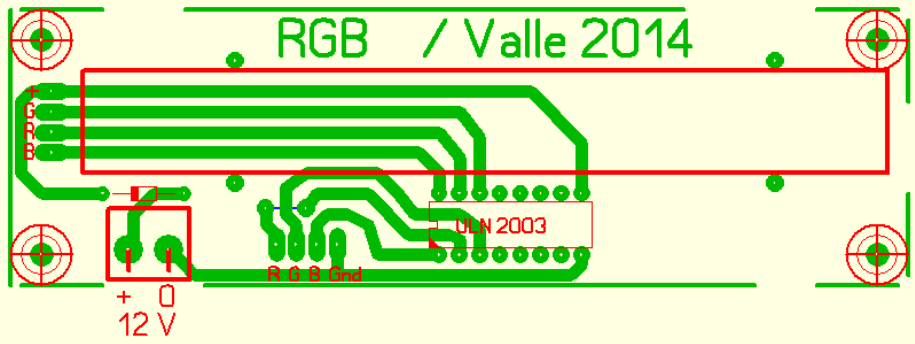
Vore øjne kan i realiteten kun opfatte rødt, grønt og blåt. Det er i hjernen, farverne mixes når der modtages signaler fra de forskellige receptorer i øjet. Dvs. ved at variere styrken af lyset fra hhv. en rød, grøn og blå lysdiode vil hjernen opfatte lyset som en anden farve.

Obs.: I stedet for RGB-båndet, kan der evt. bruges tre separate lysdioder - placeret tæt sammen. Husk formodstande!!

RGB-kittet ser således ud.

Det skal have 12 Volt forsyning fra fx en Netadapter.

Undersøg hvordan ULN2003 er opbygget !



Lysdiode-stripséne skal forsynes med 12 Volt. Brug fx en netadapter.

På kittet sidder der en ULN2003. Den fungerer som switch, dvs. hvis der er et ”1” på en indgang kan udgangen trække strøm ned til nul, selv fra 12 Volt.

Arduinoen skal styre indgangene på kittet. Et højt – eller ” 1 ” på input R, G eller B tænder de respektive lysdioder i strippen, Røde, Grønne eller Blå.

Husk også at forbinde Gnd mellem kittet og Arduino-boardet.

Lav først et program, der tænder dioderne på skift. Herefter eksperimenteres med at tænde kombinationer!

Pulsbreddemodulering af RGB-dioderne:

Hvis man tænder og slukker - dvs. pulser - en lysdiode med en høj nok frekvens, kan øjet ikke nå at registrere, at den blinker. Frekvensen skal bare være mere end ca. 25 Hz.

Men selvfølgelig vil lysmængden dioden udsender blive begrænset, hvis dioden fx kun er tændt halvdelen af tiden. Men hvis lysdioden fx er tændt i ¾ af tiden, vil den lyse kraftigere.

```
/*
Kodeeksempel til at styre RGB-dioder.
Kode fra Adafruit Arduino
Redigeret af:
Dato:

*/

int redPin = 11;
int greenPin = 10;
int bluePin = 9;

void setup()
{
  pinMode(redPin, OUTPUT);
  pinMode(greenPin, OUTPUT);
  pinMode(bluePin, OUTPUT);
}

void loop()
{
  setColor(255, 0, 0);  // red - Kald subroutine !!
  delay(1000);
  setColor(0, 255, 0);  // green
  delay(1000);
  setColor(0, 0, 255);  // blue
  delay(1000);
  setColor(255, 255, 0);  // yellow
  delay(1000);
  setColor(80, 0, 80);  // purple
  delay(1000);
  setColor(0, 255, 255);  // aqua
  delay(1000);
}

void setColor(int red, int green, int blue)
{
  analogWrite(redPin, red);
  analogWrite(greenPin, green);
  analogWrite(bluePin, blue);
}
```

Lav koden om, så farverne skifter roligt og stille

Justerbar farve med trykknapper:

Først skal der laves et program, hvor man på 3 knapper kan justere dutycyclen i de tre dioder

Hvis én knap trykkes, sendes tre forskellige værdier til RGB-kittet. Hvis en anden er trykket, 3 andre værdier osv.

```
analogWrite(pwm_pin1, value1); // Value kan have værdien fra 0 til 255.
analogWrite(pwm_pin2, value2);
analogWrite(pwm_pin3, value3);
```

Automatisk Farveskift:

En kode med brug af for-loops til pulsbreddemodulering kunne se således ud:

```
// Fade an LED using a PWM pin

int PWMpin = 10; // LED-kit or LED in series with 470 ohm resistor on pin 10

void setup()
{
  // no setup needed
}

void loop()
{
   for (int i=0; i <= 255; i++){
      analogWrite(PWMpin, i);
      delay(10);
   }
}
```

Et fadning eksempel mere:

```
// Fadning af lysdioder. Brug af analogWrite,

// Værdi fra 0 til max 255 !!!


// the setup function runs once when you press reset or power the board
```

```
  int PWMpin = 11; // choose the PWMpin for the LED

void setup() {
}
  pinMode(ledPin_1, OUTPUT);  // declare LED as output


void loop()
{
   int x = 1;
   for (int i = 0; i > -1; i = i + x){
      analogWrite(PWMpin, i);
      if (i == 255) x = -1;             // switch direction at peak
      delay(10);
   }
}
```

Det er egentligt forkert at kalde udgangene for analoge, fordi de i virkeligheden er pulsbreddemodulerede. Dvs. at de er høje en procentdel af tiden.

Lav nu flere for loops, så alle værdier på både Rød, Grøn og Blå LED vises !!!

Tip: Brug Nested For-loops

```
for (int x = 0; x < 8; x++) {
  for (int y = 0; y < 8; y++) {

    // Do something;

    delay(100);
  }
}
```

Og endnu et eksempel:

```
/*

  This sketch fades LEDs up and down one at a time on digital pins 2 through
13.
  This sketch was written for the Arduino Mega, and will not work on other
boards.

  The circuit: - LEDs attached from pins 2 through 13 to ground.

  created 8 Feb 2009 by Tom Igoe

  This example code is in the public domain.
```

```
  http://www.arduino.cc/en/Tutorial/AnalogWriteMega
*/

// These constants won't change. They're used to give names to the pins used:
const int lowestPin = 2;
const int highestPin = 13;


void setup() {   // set pins 2 through 13 as outputs:

  for (int thisPin = lowestPin; thisPin <= highestPin; thisPin++) {
    pinMode(thisPin, OUTPUT);
  }
}

void loop() {   // iterate over the pins:
  for (int thisPin = lowestPin; thisPin <= highestPin; thisPin++) {
    // fade the LED on thisPin from off to brightest:
    for (int brightness = 0; brightness < 255; brightness++) {
      analogWrite(thisPin, brightness);
      delay(2);
    }
    // fade the LED on thisPin from brightest to off:
    for (int brightness = 255; brightness >= 0; brightness--) {
      analogWrite(thisPin, brightness);
      delay(2);
    }
    // pause between LEDs:
    delay(100);
  }
}
```

Top ↑

Knightrider

Brug 8 lysdioder på et Kit. De skal lyse som Knightrider.

Kopier følgende program ind i IDE og afprøv.

```
/*
Knightrider
 Startet d. 28/1-14 @ Valle
 */

// Def af pins
int led7 = 13;
int led6 = 12;
int led5 = 11;
int led4 = 10;
```

```
int led3 = 9;
int led2 = 8;
int led1 = 7;
int led0 = 6;

int time = 100;

void setup() {                    // initialize the digital pin as an output.
    pinMode(led0, OUTPUT);
    pinMode(led1, OUTPUT);
    pinMode(led2, OUTPUT);
    pinMode(led3, OUTPUT);
    pinMode(led4, OUTPUT);
    pinMode(led5, OUTPUT);
    pinMode(led6, OUTPUT);
    pinMode(led7, OUTPUT);
}

void loop() {                     // the loop routine runs over and over again
forever:
    digitalWrite(led0, HIGH);   // turn the LED on (HIGH is the voltage level)
    delay(time);                // wait
    digitalWrite(led0, LOW);    // turn the LED off by making the voltage LOW
    digitalWrite(led1, HIGH);
    delay(time);                // wait for a second
    digitalWrite(led1, LOW);    // turn the LED off by making the voltage LOW
    digitalWrite(led2, HIGH);
    delay(time);
    digitalWrite(led2, LOW);    // turn the LED off by making the voltage LOW
    digitalWrite(led3, HIGH);
    delay(time);
    digitalWrite(led3, LOW);    // turn the LED off by making the voltage LOW
    digitalWrite(led4, HIGH);
    delay(time);
    digitalWrite(led4, LOW);    // turn the LED off by making the voltage LOW
    digitalWrite(led5, HIGH);
    delay(time);
    digitalWrite(led5, LOW);    // turn the LED off by making the voltage LOW
    digitalWrite(led6, HIGH);
    delay(time);
    digitalWrite(led6, LOW);    // turn the LED off by making the voltage LOW
    digitalWrite(led7, HIGH);
    delay(time);
    digitalWrite(led7, LOW);    // turn the LED off by making the voltage LOW

        time = analogRead(A0)/4;    // deles med 4 fordi der læses 10 bit ( 0 til
1023 )

}
```

```

```

----------;

Lav nu programmet om, så delayet ændres for hver loop.

Brug fx:

```
time = time - 10;
if (time < 15) time = 300;
```

----------;

Lav hastigheden om, så den er afhængig af justeringen af et potmeter. Arduinoen kan læse en analog værdi mellem 0 og 5 Volt, og omforme den til et tal mellem 0 og 1023.

Brug fx:

```
time = analogRead(analogPin);   // Der læses værdier fra 0 til 1023 !
time = analogRead(0)/4;         // Analog pin 0 = A0 !!
```

```
/* Knight Rider 2, Eksempel på brug af Array.
 * --------------
 *
 * Reducing the amount of code using for(;;).
 *
Kilde:  http://www.arduino.cc/en/Tutorial/KnightRider
 *
 * (cleft) 2005 K3, Malmo University
 * @author: David Cuartielles
 * @hardware: David Cuartielles, Aaron Hallborg
 */

int pinArray[] = {2, 3, 4, 5, 6, 7};
int count = 0;
int timer = 100;

void setup(){                       // we make all the declarations at once
  for (count=0;count<6;count++) {
    pinMode(pinArray[count], OUTPUT);
  }
}

void loop() {
  for (count=0;count<6;count++) {
```

```
    digitalWrite(pinArray[count], HIGH);
    delay(timer);
    digitalWrite(pinArray[count], LOW);
    delay(timer);
   }
   for (count=5;count>=0;count--) {
    digitalWrite(pinArray[count], HIGH);
    delay(timer);
    digitalWrite(pinArray[count], LOW);
    delay(timer);
   }
}
```

```
/* Knight Rider 3,
 * --------------
 *Kilde:  http://www.arduino.cc/en/Tutorial/KnightRider

 * This example concentrates on making the visuals fluid.
 *
 * (cleft) 2005 K3, Malmo University
 * @author: David Cuartielles
 * @hardware: David Cuartielles, Aaron Hallborg
 */

int pinArray[] = {2, 3, 4, 5, 6, 7};
int count = 0;
int timer = 30;

void setup(){
   for (count=0;count<6;count++) {
     pinMode(pinArray[count], OUTPUT);
   }
}

void loop() {
   for (count=0;count<5;count++) {
    digitalWrite(pinArray[count], HIGH);
    delay(timer);
    digitalWrite(pinArray[count + 1], HIGH);
    delay(timer);
    digitalWrite(pinArray[count], LOW);
    delay(timer*2);
   }
   for (count=5;count>0;count--) {
    digitalWrite(pinArray[count], HIGH);
    delay(timer);
    digitalWrite(pinArray[count - 1], HIGH);
    delay(timer);
    digitalWrite(pinArray[count], LOW);
    delay(timer*2);
   }
}
```

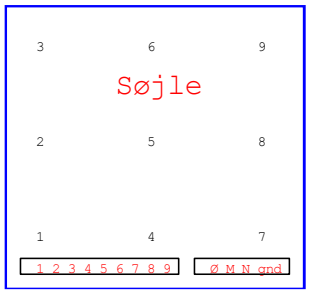
Top ↑

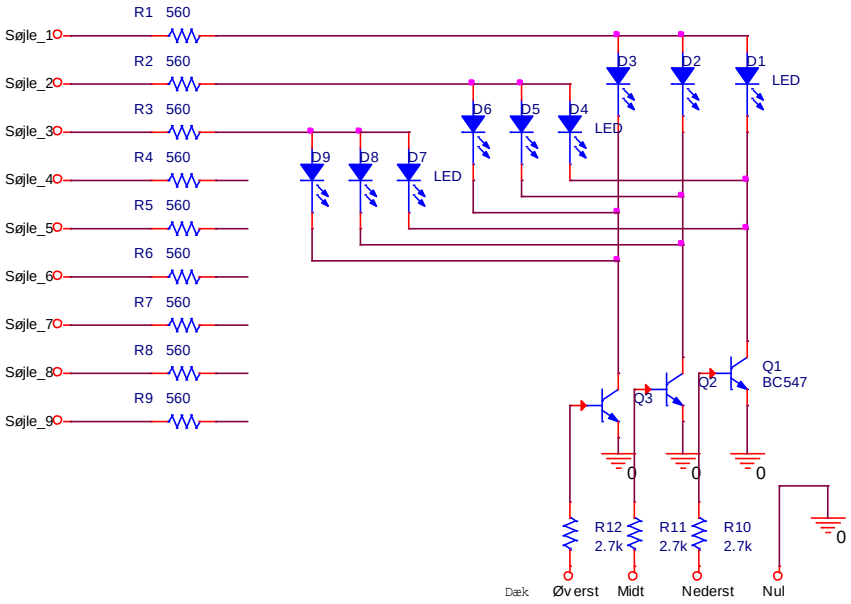
LED_Cube

Se fx denne video: https://www.instructables.com/id/Arduino-Based-3x3-LED-Cube/

Vi har et par LED-cubes, der kan styres af en Uno.

Diagrammet for dem ser således ud!





Der er 3 dæk, benævnt med Øverst, Midterste & Nederste.

Et HIGH på Ø, M eller N vælger dæk, og HIGH på pin 1 til 9 tænder en LED i pågældende søjle.

Der er brugt 3 transistorer. De kan opfattes som styrede switche. Et "1" på indgangen gennem modstanden får transistoren til at lede strøm ned til Gnd.

Her er et eksempel på et program:

```
/*
 * Et LED-Cube-program fundet derude!!!
 *
 * Valle d. 23/11-2016
 *
 */


void setup()
{
    for (int i=0;i<11;i++)
    {
```

```
        pinMode(i,OUTPUT);     // PINS 0-10 are set as output
    }
    pinMode(A0,OUTPUT);     //PIN A0 set as output
    pinMode(A1,OUTPUT);    // PIN A1 set as output
    pinMode(A2,OUTPUT);    // PIN A2 set as output

    digitalWrite(A0,LOW);     //pull up the A0 pin
    digitalWrite(A1, LOW);    // pull up the A1 pin
    digitalWrite(A2, LOW);    // pull up the A2 pin

      /* add more setup code here */
}

void loop()
{
  digitalWrite(A0,HIGH);     //layer 1 of cube is on
  for (int i=2;i<11;i++)
    {
       digitalWrite(i,HIGH);     //turn ON each LED one after another in layer1
       delay(200);
       delay(200);
       delay(200);
       digitalWrite(i,LOW);
    }
  digitalWrite(A0,LOW);     //layer1 is off

  digitalWrite(A1,HIGH);    // layer 2 of cube is grounded
  for (int i=2;i<11;i++)
    {
      digitalWrite(i,HIGH);    // turn ON each LED one after another in layer2
      delay(200);
      delay(200);
      delay(200);
      digitalWrite(i,LOW);
    }
  digitalWrite(A1,LOW);     // layer2 is off

  digitalWrite(A2,HIGH);     // layer 3 of cube is on

  for (int i=2;i<11;i++)
    {
      digitalWrite(i,HIGH);     // turn ON each LED one after another in layer3
      delay(200);
      delay(200);
      delay(200);
      digitalWrite(i,LOW);
    }
  digitalWrite(A2,LOW);     // layer3 is Off
}
```

Top ↑

Piezo Beeper

På det ene kit sidder der nogle små lydgivere.

Den ene giver lyd af sig selv, blot den forsynes med 5 Volt. Den anden skal have en pulserende spænding med den frekvens, man ønsker skal komme ud som lyd.

Der et eksempel:

```
/*
 Piezo

 This example shows how to run a Piezo Buzzer on pin 9  using the analogWrite()
function.

 It beeps 3 times fast at startup, waits a second then beeps continuously
 at a slower pace

 */

void setup()  {

  pinMode(9, OUTPUT);   // declare pin 9 to be an output:
  beep(50);
  beep(50);
  beep(50);
  delay(1000);
}

void loop()  {
  beep(200);
}

void beep(unsigned char delayms){
  analogWrite(9, 20);      // Almost any value can be used except 0 and 255
                           // experiment to get the best tone
  delay(delayms);          // wait for a delayms ms
  analogWrite(9, 0);       // 0 turns it off
  delay(delayms);          // wait for a delayms ms
}
```

Kilde:
http://www.hobbytronics.co.uk/tutorials-code/arduino-tutorials/arduino-tutorial7-piezo-beep

Top ↑

Morsekode.

```
int ledPin = 13;
void setup() // run once, when the sketch starts
{
pinMode(ledPin, OUTPUT); // sets the digital pin as output
}
void loop()
{
flash(200); flash(200); flash(200); // S
delay(300); // otherwise the flashes run together
flash(500); flash(500); flash(500); // O
flash(200); flash(200); flash(200); // S
delay(1000); // wait 1 second before we start again
}
void flash(int duration)  // subroutine. “duration” kommer fra kaldet
{
digitalWrite(ledPin, HIGH);
delay(duration);
digitalWrite(ledPin, LOW);
delay(duration);
}
```

```
/*
Program til at morse SOS
 */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
const byte ledPin = 13;  // Const -> lægges i Flash, dvs. optager ikke RAM
// #define vil lægge konstanten i RAM !!!

const int delayTime = 400;  // tid mellem karakterer
const int prikTime = 200;  // tid for prik
const int stregTime = 600;  // tid for Streg
const int pauseTime = 1800;

unsigned int count = 0;

// the setup routine runs once when you press reset:
void setup() {
    // initialize the digital pin as an output.
    pinMode(ledPin, OUTPUT);
    digitalWrite(ledPin, LOW); // der bør altid defineres en startværdi for pins
    Serial.begin(9600);
    Serial.println("ready ");
```

```
}

// the loop routine runs over and over again forever:
void loop() {

    for (int i = 0; i < 3; i++){
        digitalWrite(ledPin, HIGH);   // turn the LED on (HIGH is the voltage
level)
        delay(prikTime);               // wait for a second
        digitalWrite(ledPin, LOW);    // turn the LED off by making the voltage
LOW
        delay(delayTime);               // wait for a second
    }
    delay(delayTime);                   // pause mellem karakterer

    for (int i = 0; i < 3; i++){

        digitalWrite(ledPin, HIGH);   // turn the LED on
        delay(stregTime);               // wait for a second
        digitalWrite(ledPin, LOW);    // turn the LED off
        delay(delayTime);               // wait for a second
    }
    delay(delayTime);
    for (int i = 0; i < 3; i++){

        digitalWrite(ledPin, HIGH);   // turn the LED on
        delay(prikTime);               // wait for a second
        digitalWrite(ledPin, LOW);    // turn the LED off
        delay(delayTime);               // wait for a second
    }
    count++;
    Serial.print("antal ");
    Serial.println(count);
    delay(pauseTime);
}
```

Lav et program, der sender morsekode. Programmet skal udformes så selve lyd-genereringen ligger i hver sin subrutine.

```
//Morse SOS på pin 13

int pin = 13;

void setup()
{
```

```
  pinMode(pin, OUTPUT);

}

void loop()
{
  dot(); dot(); dot();
  dash(); dash(); dash();
  dot(); dot(); dot();
  delay(3000);
}

void dot()
{
  digitalWrite(pin, HIGH);
  delay(250);
  digitalWrite(pin, LOW);
  delay(250);
}

void dash()
{
  digitalWrite(pin, HIGH);
  delay(1000);
  digitalWrite(pin, LOW);
  delay(250);
}
```

Top ↑

Stepmotorstyring

Her skal der bruges et kit med en stepmotor. Den har nogle transistorer monteret i en IC, en ULN2003 til at switche spolestrømmen.

Motoren skal have en ekstern 12 Volt, evt. fra en netadapter.

I første omgang skal motoren bare køre.
Brug fx delay( ) - funktionen til at bestemme varigheden af de enkelte spolers energisering.

Dernæst kan der eksperimenteres med Half Step, og energisering af to spoler samtidig, så der opnås større kraft.

Lav en tæller, der tælles 1 op for hver stepmotorcyklus. Når tælleren når op på fx 300, skal motoren stoppe i 5 sekunder, hvorefter den kører modsat vej.

(Ps. Kittet er modificeret fra et ældre kit til 8051-familien.)

Lad nu delay-tiden være afhængig af en spænding, læst fra potentiometeret.

(Det er endnu ikke muligt at arbejde med eksakte tider, da dette kræver at der kører en timer sideløbende med loop-programmet. Det kræver kendskab til interrupts. Herom senere)

Stepmotorstyring:

Der skal laves et program, der kan drive en stepmotor. Start med at vise lys i 4 LED.
Der skal kunne vælges forskellige modes:
Full step, Half step, Full Step med 2 energiserede spoler samtidigt.
Koden til de forskellige modes skal lægges ned i en funktion, og funktionerne skal ligge i hver deres tab.

Outputpins skal defineres i et for-loop som henter pinnumre i et array. Arrayet bruges også af de forskellige modes.

2 ? Input-pins vælger Mode. Og -
1 Input-pin vælger omdrejningsretning
Delay defineres i en var så den kan ændres

Et potentiometer kan tilsluttes til et analog input, så en læsning af det kan bruges til at bestemme varigheden for energiseringstiden.

Lav Flowchart & Pseudokode !!

Top ↑

LCD .

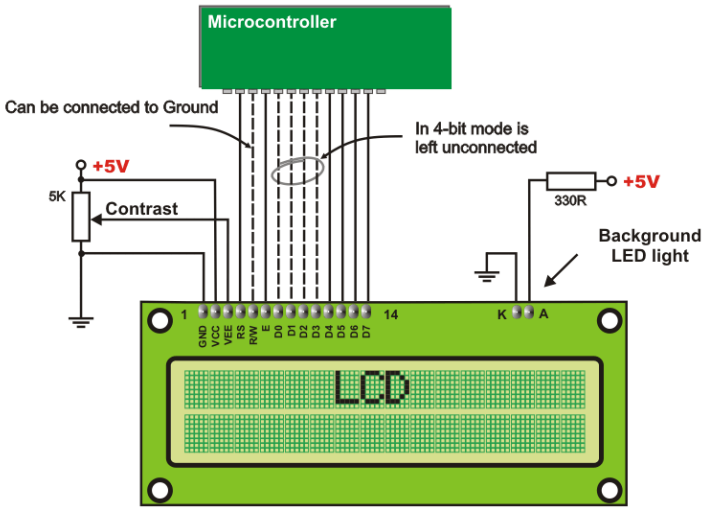
På mine kits er der monteret et LCD-display. Der er lavet de nødvendige forbindelser, der skal til for at få det til at køre.

Det viste potentiometer, vist til højre her, der skal bruges til at justere LCD-skærmens kontrast, er monteret på kittet.

Bygger man selv et print op med en LCD, skal man huske at montere et potmeter.

Med Potmeteret kan man justere spændingen på LCD-ens pin 3 mellem 0 og 5 Volt.

De stiplede ledninger D0 til D3 er udeladt på kittene.



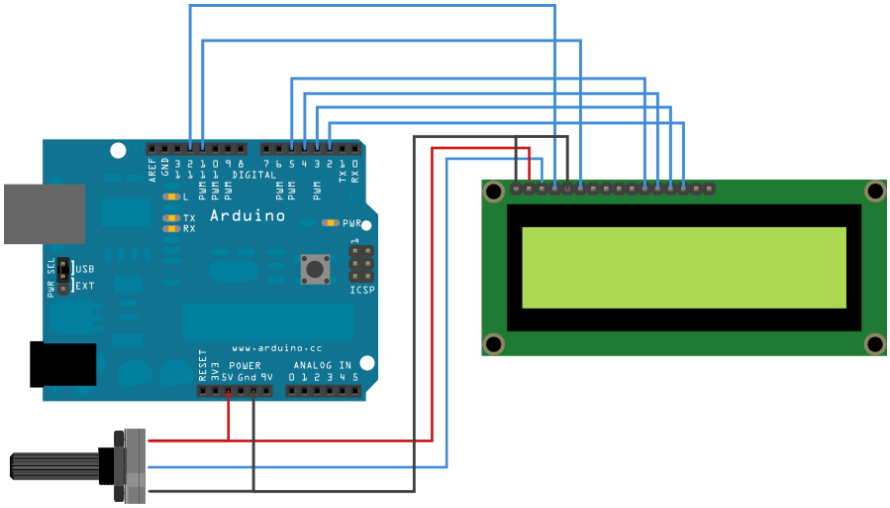
På kittene er der lavet mulighed for at tænde og slukke Backlight.

Pin 5 er forbundet direkte til Gnd. Det kan man godt, hvis man ikke har behov for at læse værdier i LCD-ens registre. og

Backlight er på LCD –en forbundet til pin 15 og 16. Backlight er kun nødvendig at bruge, hvis man skal kunne se displayet i mørke.

Muligvis er BackLight plus og minus ombyttet?

Formodstanden for Backlight bør vist ikke være mere end 10 – 15 Ohm.



Her er der et andet diagram.

Det er ikke nødvendigt med backlight i dagslys!

Pinnumre på LCD er fra venstre pin 1 til 16.

15 og 16 er til backlight.

Husk formodstand, fx 15 Ohm.

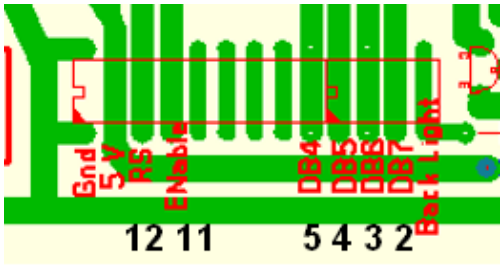
LCD-displayet kan vise almindelige bogstaver og tal, - men desværre ikke de specielle danske "æ, ø og å". I hvert fald ikke direkte.

Men der er indbygget en mulighed for selv at definere sine egne karakterer på LCD: Se fx på min hjemmeside, hvor jeg har lavet et eksempel.

Eller fx på: http://www.hackmeister.dk/2010/08/custom-lcd-characters-with-arduino/

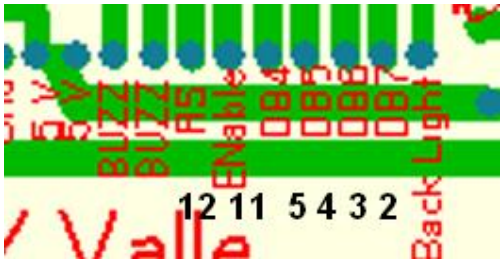
Pins fra Unoen skal forbindes til kittets stik som vist her. Men bemærk de to versioner:

Version 1



RS skal til pin Arduino pin 12, Enable til pin 11, osv.

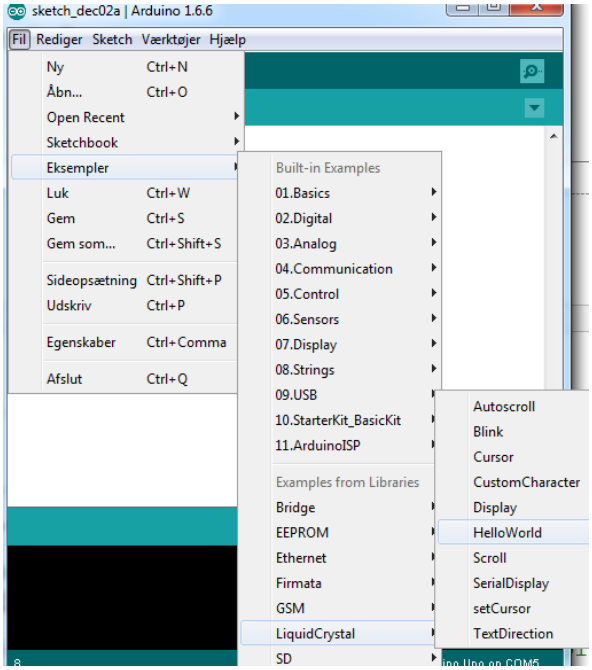
Version 2



RS skal til pin Arduino pin 12, Enable til pin 11, osv.

Opgave:

Åben koden Fil / Eksempler / LiquidCrystal / HelloWorld:



Koden vises nedenfor:

```
// include the library code:
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(20, 4);
  // Print a message to the LCD.
  lcd.print("hello, world!");
}

void loop() {
  // set the cursor to column 0, line 1
  // (note: line 0 to 3, and column 0 to 19
  lcd.setCursor(0, 1);
  // print the number of seconds since reset:
  lcd.print(millis() / 1000);
}
```

Læg mærke til, at der i kodeeksemplet er brugt en 16x2 LCD. På mine kits er der brugt LCD-er med 4 linjer a´ 20 karakterer.

Det skal derfor ændres i koden.

```
lcd.begin(20, 4);
```

Bemærk også, at der default i koden er regnet med at anvende pin 12, 11 – osv. Men de kan bare ændres, hvis det ønskes.

```
LiquidCrystal lcd(12, 11, 10, 9, 8, 7);
```

Der skal bruges mange nye funktioner i et program, når man vil arbejde med et LCD. De er smart nok lagt ned i et bibliotek, som nogen har lavet. Det skal først inkluderes i koden.

Det betyder så, at man får forærende mulighed for at Cleare skærmen, at flytte på Cursoren mm.

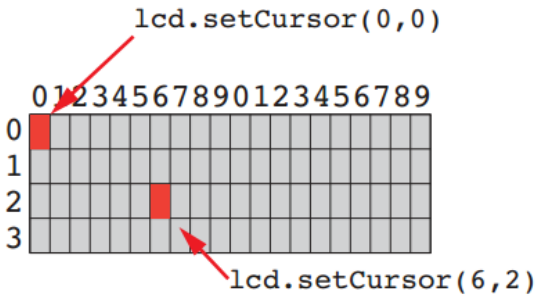
Fx:

Clear al tekst i LCD-en `lcd.clear();  // Clearer alle 4 linjer`

Placer cursoren: `lcd.setCursor(x,y);  // max 19,3`

Her er vist en funktion, der ligger i biblioteket, til at placere cursoren.

```
lcd.setCursor(x,y);
```



Arbejd nu lidt med koden. Lav om på teksten, der skrives på skærmen.

Og lav om, så der skrives på alle 4 linjer, - og efter en pause skrives 4 nye linjer tekst.

Skriv nu tekst til displayet i alle 4 linjer, efter 3 sekunder skal der skrives en ny tekst!

Byg en tæller i programmet. Bland tekst og tællerværdi på LCD-en.

Med følgende kode kan man skrive en tekst i Debugvinduet på PC-en og sende til LCD-en:.

```
void loop()
{
  // when characters arrive over the serial port...
  if (Serial.available()) {
    // wait a bit for the entire message to arrive
    delay(100);
    // clear the screen
    lcd.clear();
```

```
    // read all the available characters
    while (Serial.available() > 0) {
      // display each character to the LCD
      lcd.write(Serial.read());
    }
  }
}
```

Se: http://arduino.cc/en/Tutorial/LiquidCrystalSerial

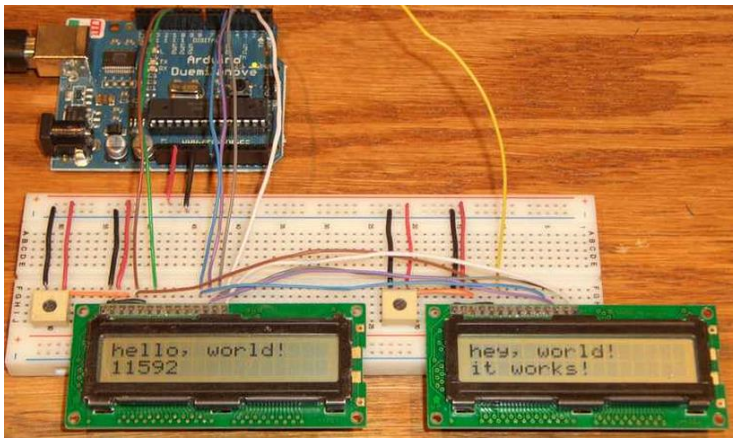
Se god youtube tutorial: http://www.youtube.com/watch?v=oIiDseJO4dM

Top ↑

Multi LCD

Det er muligt at arbejde med flere LCD-er på samme tid, og med brug af kun 1 ekstra pin
Se fx: http://www.hackmeister.dk/2010/08/4-lcd-displays-on-1-arduino/

Med blot 1 ekstra pin kan man tilslutte en LCD mere til Arduinoen. De 5 af ledningerne er fælles, det er kun Enable, der skal være separat for den LCD, man vil skrive til.



Kodeeksempel:

```
// include the library code:
#include <LiquidCrystal.h>

// initialize the library with name and the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2); // RS, Ena, 4x data.
LiquidCrystal lcd2(12, 10, 5, 4, 3, 2);

void setup() {

  lcd.begin(20, 4);              // set up the LCD's number of rows and columns:
    lcd.print("hello, world one!");     // Print a message to the 1st LCD.
    lcd2.begin(20, 4);
```

```
    lcd2.print("hey, world two!");      // Print a message to the 2nd LCD.
    lcd2.setCursor(0, 1);
    lcd2.print("it works!");
}

void loop() {
  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor(0, 1);
  // print the number of seconds since reset:
  lcd.print(millis()/1000);
}
```

Kilde: http://forum.arduino.cc/index.php/topic,5014.0.html

Egendefinerede LCD-karakterer:

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

byte smiley[8] = {
  B00000,
  B10001,
  B00000,
  B00000,
  B10001,
  B01110,
  B00000,
};
void setup() {
  lcd.createChar(0, smiley); // På plads 0 af 8 pladser ialt
  lcd.begin(20, 4);
  lcd.write(byte(0));
}
void loop() {
}
```

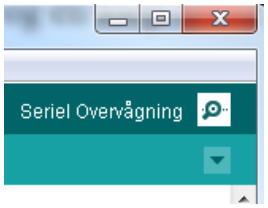
Se evt. en side med en karakter-generator her: eller her:

Se også specielt (VT) dokument om uploading danske karakterer til LCDén.

Se endvidere: http://arduino.cc/en/Reference/LiquidCrystal

Top ↑

Brug debugvinduet



Arduionoen er jo koblet til PC-en via et USB-kabel.

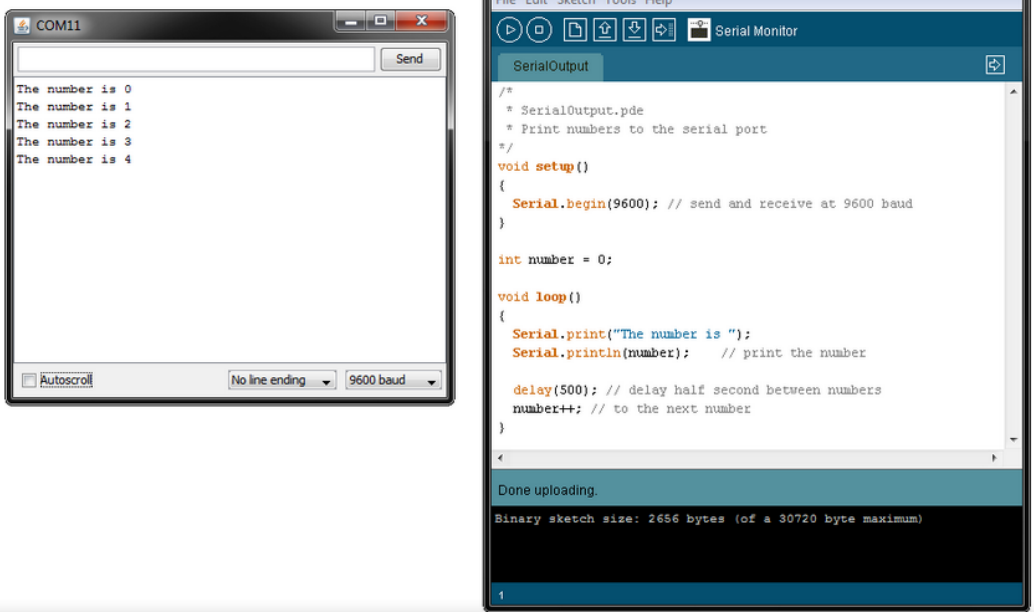
Arduinoen programmeres via kablet, men ud over dette er der mulighed for at sende data fra et program i Arduinoen til PC-en, men også fra PC-en (Keyboardet) til det program, der kører i uC-en.

I Arduino-IDE-en er der indbygget en mulighed for at åbne et monitor-vindue, der viser de data, der sendes via USB-en til PC-en.

Vinduet kaldes et Debug-vindue, da det er ret genialt til at lave debugning af et program, dvs. fejlfinding.

Man kan fx sende variables værdier til debugvinduet fra uC-programmet

Til højre er vist et sketch-eksempel, og til venstre ses debug-vinduet.



Bemærk at kommunikationen mellem UNO-en og PC-en også bruger pin 0 og 1. De kan derfor ikke bruges til fx at montere lysdioder samtidig med at USB-kablet er tilsluttet.

For at starte kommunikationen fra Arduinoen skal der i setup-program-sektionen indføjes en ordre om at medtage kode til den serielle transmission til PC-en. Derved vil compileren automatisk tilføje den nødvendig kode når kildeteksten oversættes (compileres).

Det sker i setup() som flg:

```
void setup() {
  Serial.begin(9600);             // start en seriel kommunikations-mulighed
                                  // med 9600 bit pr sekund.
}
```

Bemærk, at der skal være overensstemmelse mellem programmets baudrate og debugvinduets !!

Og data kan fx sendes som følgende:

```
Serial.print("Vaerdien er   ");    // send tekst til debugvinduet.
Serial.print(x);                   // send værdi af variabel til debugvinduet
```

Kodeeksempel:

```
int x = 0;

void  setup()
{
  Serial.begin(9600);
  Serial.println("Hello world"); // ln -> ny linje efter teksten!!
  delay(2000);// Giv tid til at se output.
}


void loop()
{
  Serial.println(x);  // Send værdien af x til PC-en
  delay(500);
  x=x+1;              // Kan også skrives: x++;
  if (x>5) {x=0;};
}
```

Se også fx Youtube: http://www.youtube.com/watch?v=T8U1CM2hkIA

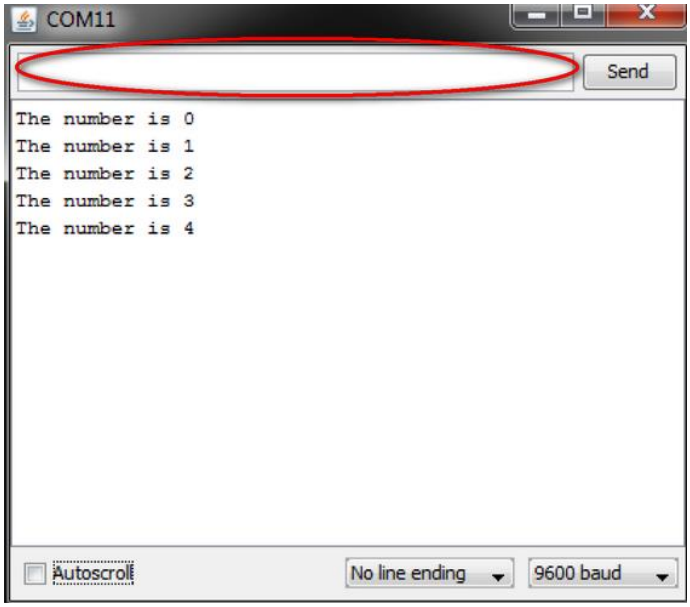
Top ↑

Send Data fra Debug-Vinduet på PC-en til Arduinoen

Ligesom man kan sende data fra Uno-en til PC-en, kan man sende data modsatte vej.

Det man vil sende, indskrives i øverste rude i Debug-vinduet, og sendes så serielt via USB-kablet.

Her er vist et eksempler på, hvordan det kan bruges:



Program-Eksempel:

```
int inByte = 0;                 // incoming serial byte
int outputPin = 13;

void setup()
{  Serial.begin(9600);          // start serial port at 9600 bps:
  pinMode(outputPin, OUTPUT);
  }

void loop()
{
  if (Serial.available() > 0) {

    inByte = Serial.read();     // get incoming byte:
    if (inByte == 'E') {
    digitalWrite(outputPin, HIGH);
    }
else if (inByte == 'F') {
    digitalWrite(outputPin, LOW);
    }
   }else{
   Serial.print('H');
  delay(1000);
  Serial.print('L');
  delay(1000);
  }
```

http://forum.arduino.cc/index.php?topic=45952.0

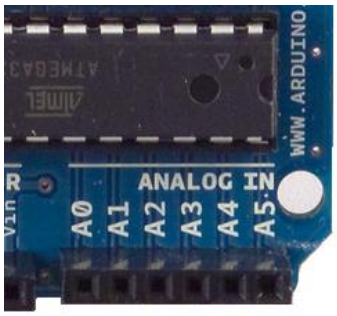
For mere: Se speciel kompendium: http://vthoroe.dk/Elektronik/Arduino/Debugvinduet.pdf

Og se fx Youtube: http://www.youtube.com/watch?v=T8U1CM2hkIA

Top ↑

Mål analog spænding

Microcontrolleren - Atmega328P - der bruges på Arduino-boardet, har indbygget mulighed for at læse analoge værdier på nogle inputs, A0 til A5



Ordren til at indlæse en værdi er

```
Value = analogRead(A0);
```

Eller

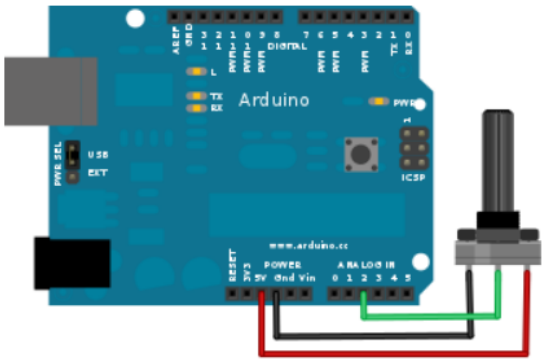
```
Variabel = analogRead(analogIndgang);
```

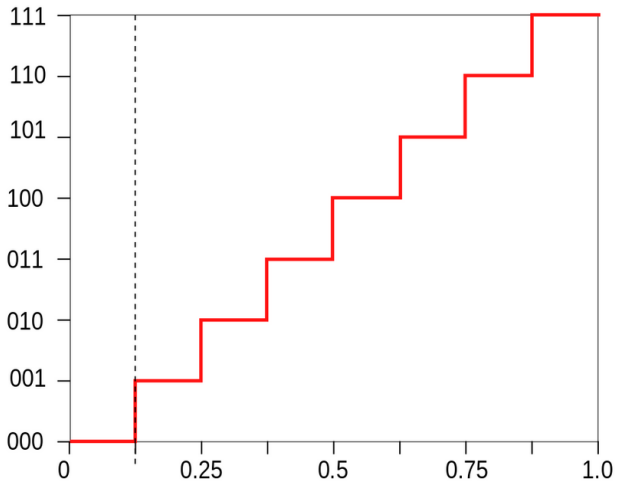
Her er vist, hvordan man kan forbinde et potentiometer til indgang A2.

Men der er også monteret et potmeter på mine kits!!

Og på nogle en LM35 temperatur-transducer.

Det kan fx også bruges til at vise temperaturen i et display.





Processoren kan jo ikke forstå analoge spændinger, kun '0' og '1' ere.

Så det, der sker, er, at den læste analoge værdi omsættes til et tal, et binært tal.

Her er vist et princip.

I Arduino uControlleren foregår det på den måde, at en analog spænding på 0 Volt omregnes til et tal med værdien 0.

Og 5 Volt omsættes til 1023.

Og derfor bliver 2,5 Volt så ca. 512.

Altså: En spænding mellem 0 og 5 Volt læses ind i en variabel, som får en tilsvarende decimal værdi mellem 0 og 1023

Skal man så udskrive den målte spænding på en skærm, er det nødvendigt at lave lidt beregning.

For det første er det vigtigt, at man vælger en variabel-type, der kan indeholde kommatal. Fx Float.

a)
Monter et potentiometer til en analog indgang, eller brug et kit. Lad den læste værdi afgøre hvor hurtigt en lysdiode blinker. Dvs. delay-længden.

b)
Den læste værdi skal også skrives på PC-skærmen i debug-vinduet. Brug Serial.Print.

c)
Hvis den læste spænding på potmeteren er lig 2,5 Volt, skal det markeres på PC-skærmen.

Hvis spændingen er > 4,5 Volt, så skal en anden LED blinke 5 gange. (Brug en For-løkke i en subrutine)

Hvis spændingen er < 0,5 Volt, skal en tredje LED lyse.

Her er eksempler på, hvad der kan bruges af kode:

```
const int analogIndgang = A0;   //Definer indgangnummer
```

```
int analogVaerdi = 0;           // definer en variable, giv den værdien 0
analogVaerdi = analogRead(analogIndgang);  // læs værdi til variabel
analogVaerdi = analogVaerdi / 4;        // omregn til max 8 bit.
```

analogVaerdi kan nu bruges i et program til fx at bestemme blinkfrekvens – eller fade-value.

En omregning kunne ske som følgende:

```
  int sensorValue = analogRead(A0);

                    // Convert the analog reading (which goes from 0 - 1023)
                    // to a voltage (0 - 5V):

  float voltage = sensorValue * (5.0 / 1023.0);

  Serial.println(voltage);       // print out the value you read:
```

Eksempel:

```
/*
  Program til at konvertere analog værdi
  og udskrive tilsvarende spænding.

*/
int sensorPin = A0;    // select the input pin for the potentiometer
float sensorValue = 0.0;  // variable to store the value coming from the
sensor

void setup() {
  Serial.begin(9600);
  Serial.println("Test af kommunikation til debug Vindue");
}

void loop() {
  sensorValue = analogRead(sensorPin);
  sensorValue = sensorValue * 5;
  sensorValue = sensorValue / 1023;
  Serial.println(sensorValue);  // Send værdien af x
  delay(100);
}
```

Et eksempel mere:

```
/*
  Arduino
```

```
  Voltmeter

*/

// Konstanter
const int analogIndgang =   A0;
const unsigned int dTime = 500;
const float gain =       204.6;

// Variabler
int analogVaerdi =           0;
float volt =               0.0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  analogVaerdi = analogRead(analogIndgang);
  Serial.print("Vaerdi fra ADC = ");
  Serial.print(analogVaerdi);

// Beregning/konvertering!

  volt = float(analogVaerdi);
  volt = volt / gain;
  Serial.print("\t Spaending = " );
  Serial.println(volt);
  delay(dTime);
}
```

Top ↑

Læs analog spænding fra Potentiometer og vis den på LCD-dispalay

```
/*
Analog pins:  Lavet af  Marcus:  1.z

Sender spændingen målt på et Potentiometer til LCD-displayet

*/

int sensorPin = A0;   // Analog pin
float sensorValue = 0.0;

#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);


void setup() {
  lcd.begin(20, 4);
}

void loop() {
```

```
  sensorValue = analogRead(sensorPin);
  sensorValue = sensorValue * 500;
  sensorValue = sensorValue / 1024;

  lcd.setCursor(2, 1);
  lcd.print("Temp er  ");
  lcd.setCursor(11, 1);
  lcd.print(sensorValue);
  lcd.setCursor(17, 1);
  lcd.print("C");
  delay(500);
}
```

Mål en temperatur

Undersøg IC-en LM35. Find fx databladet her:

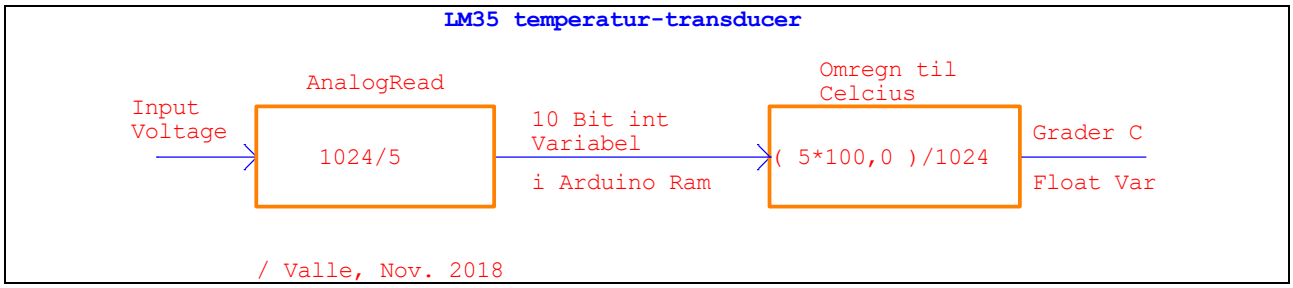
Der er monteret en LM35 temperaturføler på (nogle af) mine kits.

Lav ovenstående program om, så der måles på signalet fra temperatur-sensoren LM35.

Præsenter temperaturen i Debug-vinduet og eller på LCD-skærmen

Ps:

```
Serial.println(volt, 2);   //  Skriver 2 decimaler.
```



Se dok om ændring af intern reference her:

http://vthoroe.dk/Elektronik/AD-DA/LM35_intern%20ref_Arduino.pdf

Top ↑

Ur-program

Test følgende program!

```
/*
Ur-program

Dette program anvender et delay til at holde øje med tiden. Men det tager jo
også noget tid at udføre ordrer, så delayet skal jo ikke være 1000 mS.
*/

// Def af Konstanter

const byte ledPin =           13;
const unsigned int tDelay = 1000;   // Konstant i ROM !

// Def af Variabler

byte sekundTaeller =  55;     //  Startværdier, for test
byte minutTaeller =   59;
byte timeTaeller =    23;

byte asekund =        59;        // Alarm tidspunkt
byte aminut =         59;
byte atime =          23;

void setup()
{
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, LOW);
}

void loop()
{
  printTid();         // kald en funktion

// Indsæt din kode her!

if(sekundTaeller==asekund && minutTaeller==aminut && timeTaeller==atime) {
    digitalWrite(ledPin, HIGH);
    Serial.println("-------- ALARM! --------");
  }
  else {
    digitalWrite(ledPin, LOW);
  }

sekundTaeller++;

if (sekundTaeller >= 60) {
  sekundTaeller = 0;
  minutTaeller++;
}

if (minutTaeller >= 60) {
  minutTaeller = 0;
  timeTaeller++;
```

```
}

if (timeTaeller >= 24) {
  timeTaeller = 0;
}
  delay(tDelay);
}

//##### SUBs #####

void printTid() {
  Serial.print( "Tid: ");
  if(timeTaeller < 10) {
    Serial.print("0");
  }
  if(timeTaeller < 1) {
    Serial.print("0");
  }
  else {
  Serial.print(timeTaeller, 1);  // 1   betyder 1 decimal.
  }
  Serial.print(":");
  if(minutTaeller < 10) {
    Serial.print("0");
  }
  if(minutTaeller < 1) {
    Serial.print("0");
  }
  else {
  Serial.print(minutTaeller, 1);
  }
  Serial.print(":");
  if(sekundTaeller < 10) {
    Serial.print("0");
  }
  if(sekundTaeller < 1) {
    Serial.println("0");
  }
  else {
  Serial.println( sekundTaeller, 1);  //  sekundTaeller, 1
  }
}

/*
Syntax
Serial.print(val)
Serial.print(val, format)
*/
// ##### Ikke flere SUBs #####
```

Ovenstående program udmåler en tid vha. et delay. Men det tager jo også tid at udføre selve programmet, - så derfor kan det jo ikke være helt korrekt.

Undersøg dokumentet ”brug af millis() ” og brug dette i stedet for delay().

Men mere ”korrekt” er det nok at bruge den indbyggede interruptfunktion i uC-en.

Se specielt dokument herom – eller se senere her i ” Opgaver ”

Top ↑

Termoprinter:

Vores lille termoprinter er lavet sådan, at der kun kan skrives én linje ad gangen.

Den skal have tilsendt serielle data med 1200 bit i sekundet, - 1200 Baud.

Den første karakter, der skal sendes, er et ID, som har værdien 8Ah.

Herefter sendes et antal bogstaver som ASCII. Max ca. 10 – 12 stk.

Og sluttelig sendes en CR, en Carriage Return, eller End of String, som har værdien 0Dh.

Lav et program, der udskriver en tekst på termoprinteren. Fx hvis der trykkes på en knap
Brug SoftSerial !!

```
/*
Programeksempel til at skrive på  termoprinteren

 The circuit:
 * RX is digital pin 10 (connect to TX of other device)
 * TX is digital pin 11 (connect to RX of other device)
  */

#include <SoftwareSerial.h>

SoftwareSerial mySerial(10, 11); // RX, TX ( mySerial er blot et navn, der kan
sagtens
      // laves flere virtuelle UARTS, ex:
      // SoftwareSerial portOne(7,8);
      // SoftwareSerial portTwo(5,6);

byte rx = 10; //
byte tx = 11;

void setup()
{
  // Open serial communications and wait for port to open:
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for Leonardo only
  }
// pinMode(rx,INPUT);
  pinMode(tx,OUTPUT);
  digitalWrite(tx,HIGH);
delay(500);
```

```
  Serial.println("Hej!");     // i Debug vinduet:

  mySerial.begin(1200);   // set the data rate for the SoftwareSerial port
  delay(500);
  mySerial.write( 0x8A );   // ID for Termoprinter
  delay(100);
  mySerial.write ("Davs");     // Skriv tekst
  delay(100);
  mySerial.write( 0x0D );      // Terminate string og start skrivning
  delay(100);

 Serial.println("Sendt");     // i debugvinduet
}

void loop() // run over and over
{
}
```

Top ↑

Seriel transmission / Hej Mor-displayet.

Send serielle data til Hej Mor - Dot Matrix displayet.

Arduino Uno har kun 1 hardware-UART, og denne bruges til kommunikation med PC-en. Dvs. til upload af program, - og til kommunikation med debugvinduet.

Men der er heldigvis mulighed for meget let at tilføje ekstra ”software-UARTs” direkte fra et medfølgende bibliotek, og derved skabe en eller flere Soft-UART.

Arduino Mega har flere hardware- UARTs indbygget. Se fx. Cookbook ca. side 149.

Opgave:

Da ” Hej Mor” – displayet er bygget til 1200 Baud, skal programmet selvfølgelig vide dette.
Vælg Baudrate på 1200 Baud. (Bruges en Mega: Serial1.begin(1200);)

Vha. 4 switche tilsluttet Arduinoen vælges fx at sende forskellige tekstbeskeder.

Først sendes et ID for at vælge segment, og dernæst data.
ID er 81h for venstre digit, 82h for 2. digit osv.

```
/*
Programeksempel:

  Software serial multple serial test
 Sender til " Hej Mor " displayet
```

```
 The circuit:
 * Rx is digital pin 10 (connect to Tx of other device)
 * Tx is digital pin 11 (connect to Rx of other device)

 */
#include <SoftwareSerial.h>

byte rx = 10;       //
byte tx = 11;

SoftwareSerial hejMor(rx, tx);    // Rx, Tx er valgt til ben 10 og 11
        // I stedet for mySerial kan objektet fx kaldes ” hejMor”

void setup()
{
  // Open serial communications and wait for port to open:
  Serial.begin(9600);

// pinMode(rx,INPUT);   // er jo Input default – vist nok.
// pinMode(tx,OUTPUT);
//  digitalWrite(tx,HIGH);
  delay(500);

  hejMor.begin(1200); // indstil baudrate for SoftwareSerial port
  delay(500);

//**** Nu sendes data til displayet

  hejMor.write( 0x81 );       // ID for første segment
  delay(100);
  hejMor.write ('H');     // Skriv "H"
  delay(100);
 Serial.println("Sendt");       // i debugvinduet

  hejMor.write( 0x82 );       // ID for andet segment
  delay(100);
  hejMor.write ('e');     // Skriv "e"
  delay(100);

// osv.
}

void loop() // run over and over
{
  // do nothing
}
```

Fra venstre er ID, eller adresserne 81h, 82h … 87h.

Hvis bit 7 i en byte er sat, opfatter displayet byten som en adresse.

Ikke alle ASCII karakterer er implementeret endnu.

De specielle danske karakterer har ASCII-værdien:

æ = 01h
ø = 02h
å = 03h

Æ = 04h
Ø = 05h
Å = 06h

I protokollen er der yderligere den mulighed, at man kan sende en ID på 1010 xxxx og dernæst 7 byte data!

```
mySerial.write( 0xA0 );         // ID for  7-mode
  delay(100);
  mySerial.write ("Lagkage");           // Skriv " tekst " i display!
```

Top ↑

Selv-definerede Karakterer i Hej Mor:

Det er også muligt at skrive selvdefinerede karakterer I displayet.

Hvis der sendes en adressebyte, med Bit 4 sat høj, (fx. 10010xxxb), afventer displayet på yderligere 5 bytes med mønster-information, dvs. hvilke lysdioder, der skal tændes. Værdierne xxx er fra 1 til 7 for at vælge hvilken digit man skriver til.

Den første byte i mønsteret er venstre søjle, 2. byte er næste søjle osv.

Protokollen for de 5 mønster-data-bytes er: 0xxxxxxxb, hvor bit 6 er den øverste LED, og bit 0 er den nederste LED i søjlen. Bit 7 skal være 0, ellers tror de andre segmenter, at det er en adresse..

```
  mySerial.write( 0x91 );      // ID for selv-definerede karakterer
  delay(100);
  mySerial.write (B01110111);           // venstre søjle
  mySerial.write (B00100010);           // Binære tal kan også defineres som
                                        // 0b00100010
  mySerial.write ((byte)B00000000);     // Man kan ikke umiddelbart compilere en
                                        // byte som er 0!
  mySerial.write (B00100010);
```

```
  mySerial.write (B01110111);
```

Obs: Hvis der skal sendes en hel byte med 8 '0'-ere, skal det skrives som:

mySerial.write ((byte)B00000000); Derfor bør det smart gøres i alle linjer!

Top ↑

7-segment

Skal der laves program til et 7-segment, skal der jo konverteres fra et tal til et mønster, der på et 7-segment skriver fx et 5-tal. Der skal bruges en konverteringstabel. !!

Kodeeksempel:

```
// Denne opgave er lavet sammen med EUX 30/4-2020

/*
Tabel til at konvertere et tal, fra 0 til 9, til et 7-segment-mønster.
Aktiv 0, dvs. lav segmenterne lyser op på et 0 !!
Pinrækkefølge til 7-segmenterne:  *gfd ceab
Det indrettes sådan, at højreste bit styrer højreste segment, her segment b.

*/

// Def af mønstertabel

uint8_t tabel[10] = { // Mønstertabel
B01000000, // 0
B01110110, // 1
B00101000, // 2
B00100100,
B00010110,
B00000101,
B00000001,
B01110100,
B00000000,
B00010100};

// def af Pins til at styre hvilken 7-segment, der er aktiv

byte segv = 10;  // styrepins til multiplexing
byte segh = 9;

// def af pins til hver LED i 7-segmenterne-
// Skal da smartere laves i et Array. Så er det lettere at ændre

byte b = 2;  // det er rigtig smart, hvis de er i rækkefølge
```

```
byte a = 3;
byte e = 4;
byte c = 5;
byte d = 6;
byte f = 7;
byte g = 8;

uint8_t _10ere = 3;  // variabel til ti-erne
uint8_t _1ere = 7;   // og til enere
uint8_t x=0;

//************************************************
void setup() {

  for (int pin = 2; pin < 11; pin++) { // def outputpins
    pinMode(pin, OUTPUT);
  }

  for (int pin = 2; pin < 11; pin++) { // slukker alle segmenter
    digitalWrite(pin, HIGH);  // er jo i dette eks. Med fælles anode
  }
}
//****************************************

void loop() {

  x = tabel [_1ere];  // hent mønster for enere

  for (int i = 0; i < 8; i++) {
    digitalWrite((i + 2), bitRead(x, i)); // Hent bit og send ud
  }
  digitalWrite(segh, LOW);  // tænd strøm til højreste 7-segment
  delay(1000);
  digitalWrite(segh, HIGH);  // og sluk igen

//*********************************

x = tabel [_10ere]; // hent mønster for tiere

  for (int p = 0; p < 8; p++) {
    digitalWrite((p + 2), bitRead(x, p));
  }
  digitalWrite(segv, LOW);
  delay(1000);
  digitalWrite(segv, HIGH);


}
```

Et eksempel med et to-dimensionel bit-array. Her er der 10 hylder, med 8 lagerpladser mod højre. Dvs. nummer 0,0 er øverste venstre bit, og 10,8 er nederste højreste bit.

```
//Kode til 7-segment med brug af et 2-dimensionel - bit-Array:
```

```
bool num_array[10][8] = { // [y][x], Y er nedad, x bit fra venstre mod højre
  { 1, 1, 1, 1, 0, 1, 1, 0 }, // 0
  { 1, 0, 0, 1, 0, 0, 0, 0 }, // 1
  { 1, 1, 0, 1, 0, 1, 1, 0 }, // 2
  { 1, 1, 0, 1, 1, 1, 0, 0 }, // 3
  { 1, 0, 1, 1, 1, 0, 0, 0 }, // 4
  { 0, 1, 1, 1, 1, 1, 0, 0 }, // 5
  { 0, 1, 1, 1, 1, 1, 1, 0 }, // 6
  { 1, 1, 0, 1, 0, 0, 0, 0 }, // 7
  { 1, 1, 1, 1, 1, 1, 1, 0 }, // 8
  { 1, 1, 1, 1, 1, 1, 0, 0,}  // 9
};
int digit1 = 0;  // Enere, styres af pin 12
int digit2 = 0;  // Tiere, styres af pin 13

//*************************************

void setup()
{
  // Serial.begin(4800);

  pinMode(5, OUTPUT); // b, def pins som output. Passer ikke til vore kit.
  pinMode(6, OUTPUT); // a
  pinMode(7, OUTPUT); // f
  pinMode(8, OUTPUT); // c
  pinMode(9, OUTPUT); // g
  pinMode(10, OUTPUT); // d
  pinMode(11, OUTPUT); // e
  pinMode(12, OUTPUT); //  right digit
  pinMode(13, OUTPUT); //  left digit

  digitalWrite(12, HIGH); // Sluk segment,
  digitalWrite(13, HIGH); // Sluk segment,

}

//*************************************

void loop() {

  // ************************  Digit 1

  for (int pin = 5; pin < 12; pin++)
  {
    digitalWrite(pin, num_array[digit1][pin - 5]);
  }
  digitalWrite(12, LOW);  // Tænd segment, aktiv lav

  delay(5);
  digitalWrite(12, HIGH); // Sluk segment,

  // **************************Digit 2

  for (int pin = 5; pin < 12; pin++)
  {
    digitalWrite(pin, num_array[digit2][pin - 5]);
  }
  digitalWrite(13, LOW); // Tænd segment,
  delay(5);
```

```
  digitalWrite(13, HIGH); // Sluk

/* brug fx her millis() funktionen til udmåle 1 sekund til at update 1-ere og
10-ere.

*/

}

//Fra <https://github.com/Wozar/school-electronics/blob/master/Segment.ino>
```

Se fa kilder:

http://www.hobbytronics.co.uk/arduino-4digit-7segment
http://www.hacktronics.com/Tutorials/arduino-and-7-segment-led.html Fælles katode!
http://www.tinkerhobby.com/arduino-2-digit-7-segment-display-counter/

Top ↑

7-segment kit, Multiplexing.

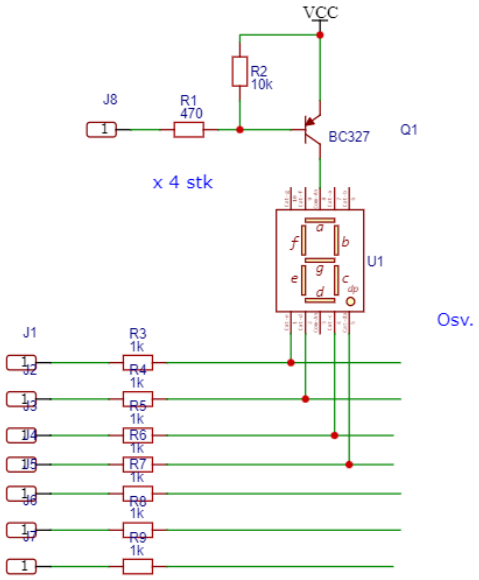
Der skal forbindes ledninger fra 7 udgange i Arduinoen til de 7 øverste klemmer med de grønne ledninger. De styrer de enkelte segmenter i 7-segmenterne, a til g. Obs: De er ikke i rigtig rækkefølge !! De er ”Aktiv lav ”

I nederste klemrække fra Venstre:
Rød = 5 Volt.
Sort = 0 Volt (Gnd) skal kun forbindes, hvis man ønsker at tænde punktummet i 2. 7-segment fra venstre !!

Så følger de 4 digitvalg, der hver vha. de 4 transistorer aktiverer strømmen til et af 7-segmenterne. De er også ” Aktiv lav ”.

Til højre er der vist et princip-diagram. Det er ikke endelig !! men viser princippet.

Lav fx kode, er viser minutter og sekunder



Top ↑

Pernille-display

Kittet ” Pernille ”, der kan vise 2 tal i 2 7-segmenter, er styret af en ATMEL 8051-familie uC.

Det er bygget så det kan modtage serielle data fra omverdenen ved 1200 Baud.

Lav et program, der tæller op på displayet. Fx hver gang der trykkes på en knap, eller få displayet til at vise fortløbende sekunder.

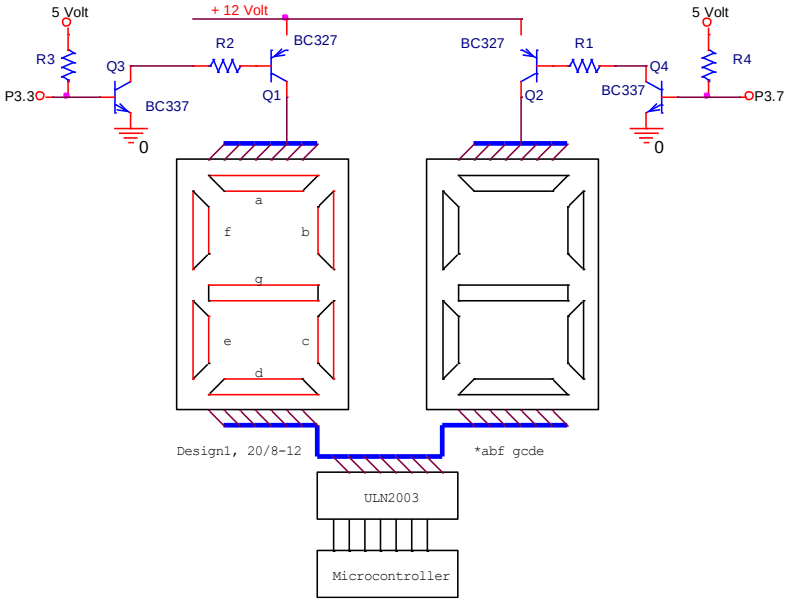
Kittes ID er 8Bh

Diagrammet viser hvordan systemet er forbundet!

Protokol:
Data skal modtages serielt!!

Der sendes først en ID = 8Bh, og dernæst to byte med først 10-ere så 1-ere i low nibble.

1200 Baud.



Obs: I stedet for mySerial kan man navngive objektet fx pernilleSerial.

Hvis den værdi, der skal sendes serielt har værdien 0, forstår compileren det ikke. Derfor skal man fortælle den, at det er byte-værdien af et tal, der skal sendes.
Eks:

```
pernilleSerial.write ((byte)i);  // (byte skal skrives fordi 00h ikke kendes af
                                 // compileren.
```

```
/*
Programeksempel til at sende til Pernillekittet.

 Der skal sendes en ID = 0x8B efterfulgt af to bytes, tal fra 0 til 9
 * RX is digital pin 10 (connect to TX of other device)
 * TX is digital pin 11 (connect to RX of other device)

 */
#include <SoftwareSerial.h>

SoftwareSerial mySerial(10, 11); // RX, TX    ( kald evt. mySerial for
pernilleSerial i stedet )

byte rx = 10; //  Pins for RxD, Recieve Data,
byte tx = 11; //   Pin for  TxD, Transmit Data.

void setup()
{
    Serial.begin(9600); // Open serial communications and wait for port to open:
    while (!Serial) {
        ; // wait for serial port to connect. Needed for Leonardo only
    }
```

```
    // pinMode(rx,INPUT);
    pinMode(tx,OUTPUT);
    digitalWrite(tx,HIGH);
    delay(500);
    Serial.println("Hej!");    // i Debug vinduet:
    mySerial.begin(1200);    // set the data rate for the SoftwareSerial port
    delay(500);
}

void loop() // run over and over
{
    mySerial.write( 0x8B );   // ID for Pernille
    mySerial.write (0x03);    // 10ere sendes
    mySerial.write (0x07);    // 1ere sendes
    delay(2000);
    //
    mySerial.write( 0x8B );   // ID for Pernille
    mySerial.write (0x09);
    mySerial.write (0x05);    //
    delay(2000);
    //
}
```

Der er et problem med at sende en seriel 0x00

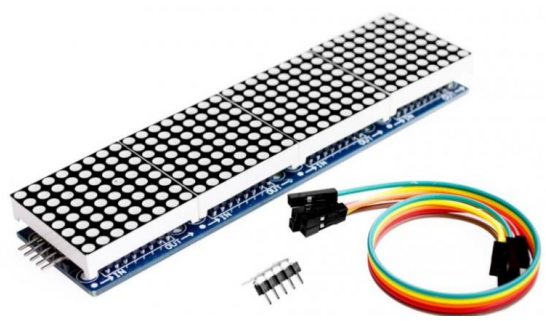
```
/*
 Programeksempel til at sende til Pernillekittet.

 Der skal sendes en ID = 8B efterfulgt af to bytes, tal fra 0 til 9
 * RX is digital pin 10 (connect to TX of other device) !! Vigtigt!
 * TX is digital pin 11 (connect to RX of other device) !! Vigtigt!

 Revideret af Daniel Munk: 13-11-2013

 */

#include <SoftwareSerial.h>

SoftwareSerial pernilleSerial(10, 11); // RX, TX ( kald evt. mySerial for
pernilleSerial i stedet )

byte rx = 10; // Pins for RxD, Recieve Data,
byte tx = 11; // Pin for TxD, Transmit Data.
int i = 1;    // Variabel for 10'ere
int t = 1;    // Variabel for 1'ere

void setup()
```

```
{
  Serial.begin(9600); // Open serial communications and wait for port to open:
  while (!Serial) {
    ; // wait for serial port to connect. Needed for Leonardo only
  }
  // pinMode(rx,INPUT);
  pinMode(tx,OUTPUT);
  digitalWrite(tx,HIGH);
  delay(500);
  Serial.println("Hej!"); // i Debug vinduet:
  pernilleSerial.begin(1200); // set the data rate for the SoftwareSerial port
  delay(500);
}

void loop() // run over and over
{
  for (i = 0x00; i <=0x09; i++) // 10'erne (i) skal starte i 0, tælle op til den
er 9 og sættes til 0 igen
  {
    for (t = 0x00; t <=0x09; t++) // 1'erne (t) skal starte i 0, tælle op til den
er 9 og sættes til 0 igen (t-for-loopet gør, at loopet kan gå videre til i-for-
loopet hvis den kommer over 9)
    {

      pernilleSerial.write( 0x8B ); // ID for Pernille
      pernilleSerial.write ((byte)i); // 10ere sendes (byte skal skrives fordi
00h ikke kendes)
      pernilleSerial.write ((byte)t); // 1ere sendes (byte skal skrives fordi 00h
ikke kendes)
      delay(1000);
    }
  }

}
```

Top ↑

Dot-Matrix display

Der skal opbygges et system, der kan multiplexe et 5x8 dot matrix display.



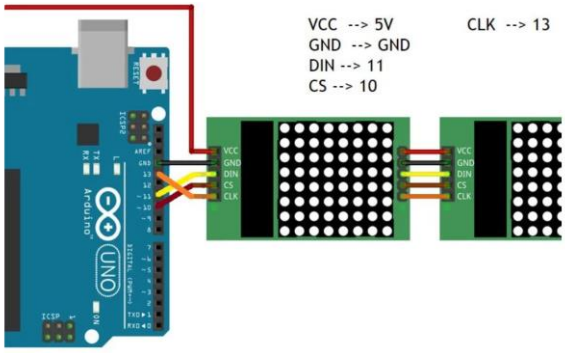
Skriv / test et dot matrix display.

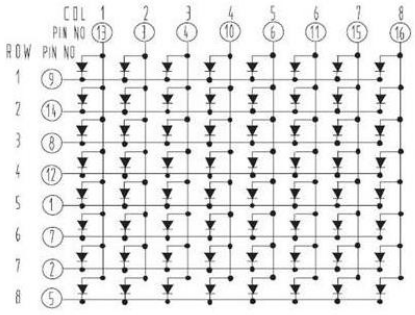
Under selve displayet sidder en MAX7219 IC.

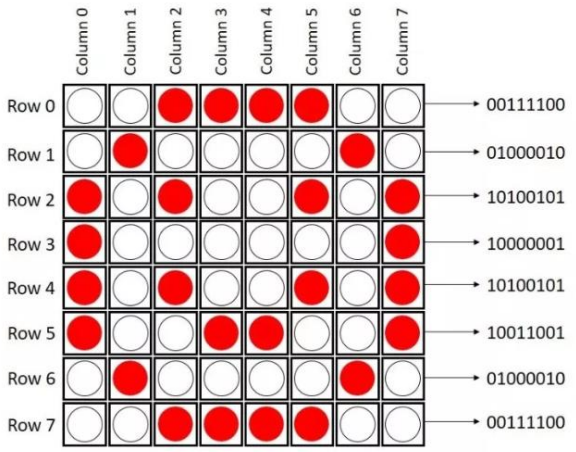
Den skal undersøges.

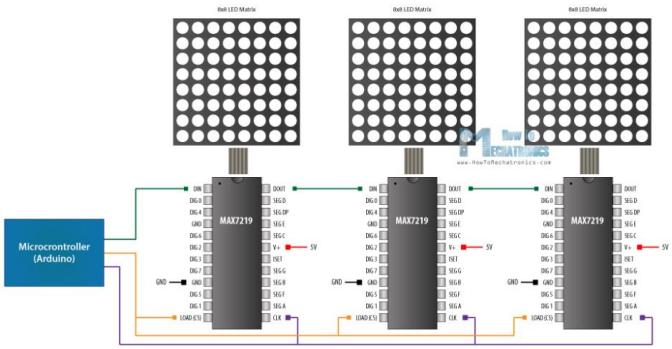
Forbindelser:

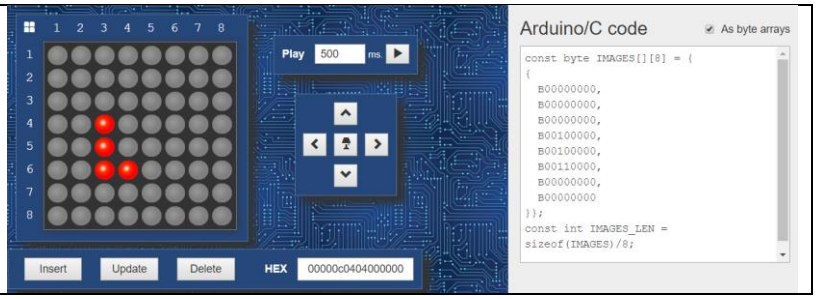
VCC --> +5V
GND --> GND
DIN (DATA PIN) -->11
CS PIN --> 10
CLK PIN --> 13









Side med smart kode-generator !!	

Se: https://xantorohara.github.io/led-matrix-editor/
Der er en anden på: http://dotmatrixtool.com/

Denne kode virker: https://www.brainy-bits.com/how-to-control-max7219-led-matrix/

Nedenstående kodeeksempel stammer herfra:

```
// Ikke testet !!

// Fra:

/*
  Basic code for using Maxim MAX7219/MAX7221 with Arduino.

  Wire the Arduino and the MAX7219/MAX7221 together as follows:

  | Arduino    | MAX7219/MAX7221 |
  | ---------- | --------------- |
  | MOSI (11)  | DIN             |
  | SCK (13)   | CLK             |
  | I/O (7)*   | LOAD/CS         |

    * - This should match the LOAD_PIN constant defined below.

  For the rest of the wiring follow the wiring diagram found in the datasheet.

  Datasheet: http://datasheets.maximintegrated.com/en/ds/MAX7219-MAX7221.pdf

  Author:  Nicholas Dobie <nick@nickdobie.com>
  Date:    30 December 2013
  License: WTFPL (http://www.wtfpl.net/)
 */
#include <SPI.h>

// What pin on the Arduino connects to the LOAD/CS pin on the MAX7219/MAX7221
#define LOAD_PIN 7

/**
 * Transfers data to a MAX7219/MAX7221 register.
 *
 * @param address The register to load data into
 * @param value   Value to store in the register
 */
void maxTransfer(uint8_t address, uint8_t value) {

  // Ensure LOAD/CS is LOW
  digitalWrite(LOAD_PIN, LOW);

  // Send the register address
  SPI.transfer(address);

  // Send the value
  SPI.transfer(value);

  // Tell chip to load in data
```

```
  digitalWrite(LOAD_PIN, HIGH);
}


void setup() {

  // Set load pin to output
  pinMode(LOAD_PIN, OUTPUT);

  // Reverse the SPI transfer to send the MSB first
  SPI.setBitOrder(MSBFIRST);

  // Start SPI
  SPI.begin();

  // Run test
  // All LED segments should light up
  maxTransfer(0x0F, 0x01);
  delay(1000);
  maxTransfer(0x0F, 0x00);

  // Enable mode B
  maxTransfer(0x09, 0xFF);

  // Use lowest intensity
  maxTransfer(0x0A, 0x00);

  // Only scan one digit
  maxTransfer(0x0B, 0x00);

  // Turn on chip
  maxTransfer(0x0C, 0x01);



}

void loop() {

  // Loop through each code
  for (uint8_t i = 0; i < 0x10; ++i)
  {
    maxTransfer(0x01, i);
    delay(500);
  }

}
```

Et andet kodeeksempel her:

Se også specielt dokument om dot matrix displays:

Top ↑

RF-ID

RF-ID står for Radio Frequencies IDentification. Se dokument herom:

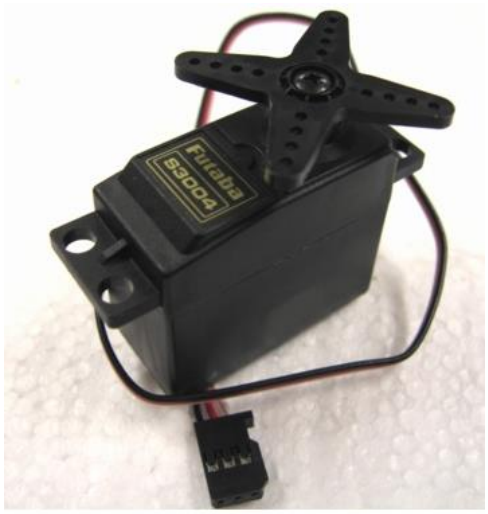
Lav et program, der læser RF-ID tags og skriver deres ID i Debug-vinduet på PC-skærmen, eller på et LCD-display.

Top ↑

RC-Servomotor

En RC-Servomotor er en lille motor, der kan styres til at dreje til en ønsket vinkel, og forblive her, indtil den fx skal dreje til en anden vinkel.

Normalt kan en RC-Servomotor dreje maximalt ca. 180 grader fra side til side, men der findes også motorer, der kan rotere kontinuerligt.

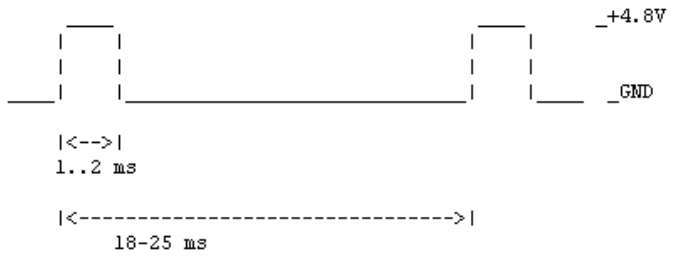


Benforbindelser:

BLACK Ground
WHITE Control pin
RED +4.8V power supply (+5V works well)

Motoren styres af nogle pulser, som sagtens kan komme fra Arduinoen !!

Se info om RC-Servomotorer her:

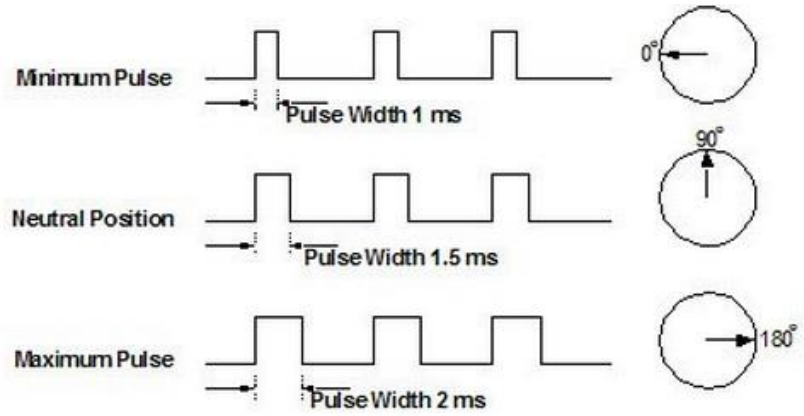


Vha. bredden på pulserne på control-ledningen kan man bestemme motorens position.

Pulserne skal gentages hver ca. 20 mS.

Motorens position styres af pulsbredden på signalledningen.

Motoren drejer fra den ene yderstilling til den anden hvis pulsernes bredde ændres fra 1mS til 2mS.



Brug det medfølgende Servo-motor-bibliotek til at styre servoer direkte.

I kan opbygge jeres eget system, eller som en start bruge mit Servokit. Det skal have data tilsendt serielt:

Baud 1200
ID = 8Ch + 4 bytes.
Motor 0 er venstre i bunden
Motor 1 er venstre på armen
Motor 2 er højre på armen
Motor 3 er højre i bunden (set fra uC-en.)

Data sendes som 10001100 xxxxxxxx xxxxxxxx xxxxxxxx xxxxxxxx
Hvor xxxxxxxx er værdier fra 1d til 180d.

Data er: ID, Motor0, Motor1, Motor2, Motor3, Motor4

Servoerne starter med at bevæge sig ved Power_on.

Men så snart der ankommer serielle data, går servoens uC over i et nyt program, der adlyder de modtagne komandoer.

Hvis P3.5 er lav, hoppes direkte til styring kun med serielle data
Hvis P3.4 er lav, drejer alle motorer fra 0 til 180 grader!!

Top ↑

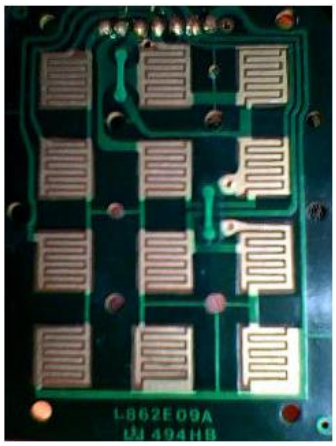
Keypad

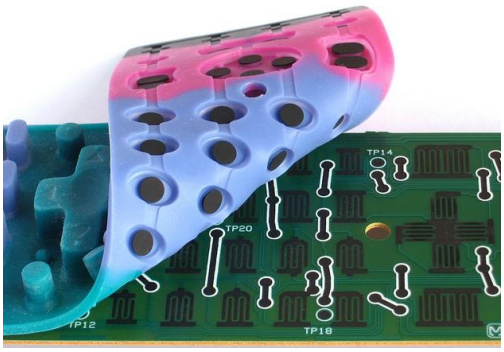
Forbind et Keypad til Arduinoen, - og lav et program, der fx skriver i debugvinduet på PC-en eller på en LCD-skærm, hvilken tast, der trykkes.

Hvordan virker et Keypad?

Hvis et keypad ikke er konstrueret som et matrix, skulle der til et 4x4 keypad bruges en plus, og 16 ledninger til Arduinoen. Dette ville næsten bruge alle pins, - så derfor bruges normalt udelukkende matrix-typer.

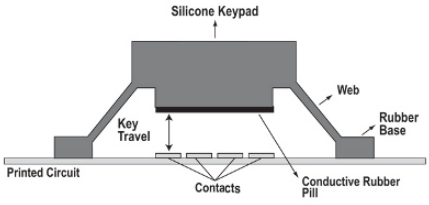
Et keypad kan typisk være konstrueret som vis på disse billeder.





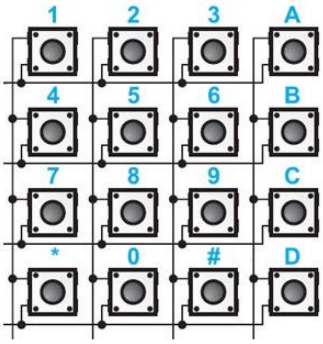
Når tasten trykkes, presses et ledende gummimateriale ned på et print og kortslutter to ledninger.

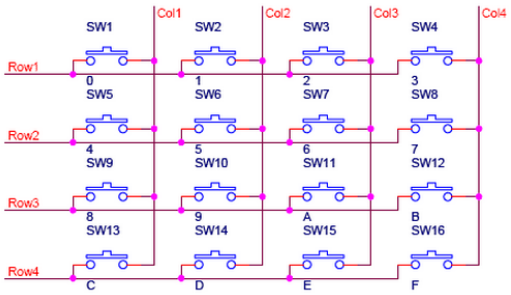
Her er en kontakt vist i skitse



Her et keypad med matrix forbindelser.

Der er 4 rækker og 4 søjler. Og et tryk giver en forbindelse mellem en søjle og en række.



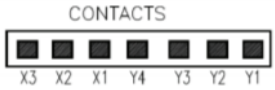


Vær opmærksom på, at søjler og rækker er vist fra bagsiden i databladet.

Venstre Søjle er kolonne, S1. - Øverste række er R1.

Pins på 4x4 Keypad set forfra er fra venstre:

R1, R2, R3, R4 – S1, S2, S3



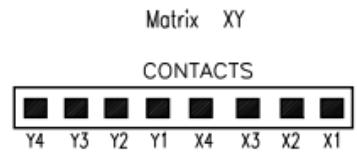
Obs.: Set fra bagsiden !!

	X1	X2	X3
Y1	1	2	3
Y2	4	5	6
Y3	7	8	9
Y4	*	0	#

Farnell Varenummer: 1130805

Pins på 4x4 Keypad set forfra er fra venstre:

S1, S2, S3, S4, - R1, R2, R3, R4



Obs.: set fra bagsiden !

	X1	X2	X3	X4
Y1	1	2	3	F
Y2	4	5	6	E
Y3	7	8	9	D
Y4	A	0	B	C

Farnell Varenummer: 1130806

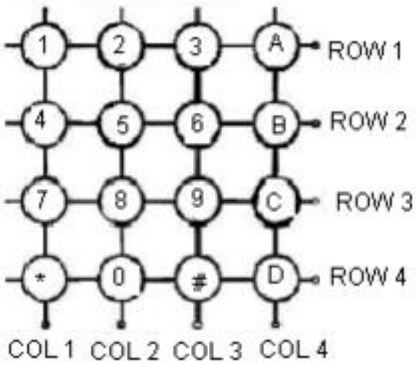
Se: https://www.farnell.com/datasheets/2010030.pdf

Output Arrangement	
Output Pin Number	Symbol
1	COL 1
2	COL 2
3	COL 3
4	COL 4
5	ROW 1
6	ROW 2
7	ROW 3
8	ROW 4

Når der trykkes på en tast, vil der opstå en kortslutning mellem en ledning fra kolonnerne og fra rækkerne.

Nu er det bare processorens opgave at aflæse hvilken det er, - men det er bare ikke så let.

Det kan foregå efter følgende procedure:



Processoren skal fx sætte HIGH på alle rækker, og definere alle søjler som inputs.

Derefter skal der tjekkes om alle søjler er lave. Hvis, er der ikke trykket på en tast.

Trykkes der en tast – vil en af søjlerne blive høje. – Men man kan jo ikke vide hvilken knap i søjlen det er. Det skal nu tjekkes.

Det kan ske ved kun at gøre én række høj af gangen, og for hver gang tjekke søjlen. På den måde kan man finde frem til den rigtige knap, der er trykket.

På alle pins er der på kittene monteret en 10Kohm pulldown modstand.

```
//Programeksempel - Uden brug af et bibliotek. Et Lineært program:


/*Program demonstrating 4x4 Numeric Keypad interfacing with Arduino UNO
Program Written by: Amit Biswal.  Se kilde:
 http://www.123mylist.com/2012/12/4x4-keypad-interfacing-with-arduino-uno.html

Modificeret af Valle Thorø, d. 02/11-2016 til vore testkits

Pins på 4x4 Keypad set forfra er fra venstre:
```

```
C1, C2, C3, C4, R1, R2, R3, R4

Husk at der også skal forbindes et Gnd til kittet fordi alle 8 keypads-pins
har Pull Down modstande.

Bemærk også, at programmet godt kan nå at gennemløbe flere omgange ved 1
tastetryk
og at programmet godt lige kan have passeret test af den knap man trykker,
hvorfor der vil opleves et delay.
*/

// Def af pins. Bemærk, at de analoge input pins er brugt !!

int r1=A5;     // Række 1 Øverste række ???
int r2=A4;
int r3=A3;     // R = Row
int r4=A2;

int c1=A1;     // Søjle 1  Venstre søjle ???
int c2=A0;     // c = Colmn
int c3=10;
int c4=11;

void setup()
{
  Serial.begin(9600);   // For test på debugskærm

  pinMode(r1,OUTPUT);   // Definer rækker som output
  pinMode(r2,OUTPUT);
  pinMode(r3,OUTPUT);
  pinMode(r4,OUTPUT);

  pinMode(c1,INPUT);    // Definer søjler som input.
  pinMode(c2,INPUT);
  pinMode(c3,INPUT);
  pinMode(c4,INPUT);

}
void loop()
{
  int val;
  //setting the columns as high initially
 // digitalWrite(c1,HIGH);
 // digitalWrite(c2,HIGH);
 // digitalWrite(c3,HIGH);
 // digitalWrite(c4,HIGH);

  //checking everything one by one
  //case 1: row1 =1 while other col is 0
  digitalWrite(r1,HIGH);
  digitalWrite(r2,LOW);
  digitalWrite(r3,LOW);
  digitalWrite(r4,LOW);

  //checking each column for row1 one by one
  if(digitalRead(c1)==1)  // Tjek for høj
  {
    Serial.println("key 1 pressed");
```

```
  }
  else if(digitalRead(c2)==1)
  {
    Serial.println("Key 2 pressed");
  }
  else if(digitalRead(c3)==1)
  {
    Serial.println("Key 3 pressed");
  }
  else if(digitalRead(c4)==1)
  {
    Serial.println("Key F pressed");
  }

  //case 2: row2 =1 while other col is 0
  digitalWrite(r1,LOW);
  digitalWrite(r2,HIGH);
  digitalWrite(r3,LOW);
  digitalWrite(r4,LOW);

  //checking each column for row2 one by one
  if(digitalRead(c1)==1)
  {
    Serial.println("key 4 pressed");
  }
  else if(digitalRead(c2)==1)
  {
    Serial.println("Key 5 pressed");
  }
  else if(digitalRead(c3)==1)
  {
    Serial.println("Key 6 pressed");
  }
  else if(digitalRead(c4)==1)
  {
    Serial.println("Key E pressed");
  }

  //case 3: row3 =1 while other crow is 0
  digitalWrite(r1,LOW);
  digitalWrite(r2,LOW);
  digitalWrite(r3,HIGH);
  digitalWrite(r4,LOW);

  //checking each column for row3 one by one
  if(digitalRead(c1)==1)
  {
    Serial.println("key 7 pressed");
  }
  else if(digitalRead(c2)==1)
  {
    Serial.println("Key 8 pressed");
  }
  else if(digitalRead(c3)==1)
  {
    Serial.println("Key 9 pressed");
  }
  else if(digitalRead(c4)==1)
  {
```

```
    Serial.println("Key D pressed");
  }

  //case 4: row4 = 1 while other row is 0
  digitalWrite(r1,LOW);
  digitalWrite(r2,LOW);
  digitalWrite(r3,LOW);
  digitalWrite(r4,HIGH);

  //checking each column for row4 one by one
  if(digitalRead(c1)==1)
  {
    Serial.println("key A pressed");
  }
  else if(digitalRead(c2)==1)
  {
    Serial.println("Key 0 pressed");
  }
  else if(digitalRead(c3)==1)
  {
    Serial.println("Key B pressed");
  }
  else if(digitalRead(c4)==1)
  {
    Serial.println("Key C pressed");
  }
  //giving delay between keypress
  delay(200);

}
```

Kode-Eksempel Keypad

```
/*
Programeksempel, hvor der er brugt løkker, - og selve test af keypad sker ved
kald til en subrutine

Keypad sketch
 prints the key pressed on a keypad to the serial port
 Modificeret d. 3/11-2013 by Valle
 Til Keypads med Pull Down-modstande.

Pins: set forfra, fra venstre: Kolonne / Søjle K0, K1, K2, (K3), R0, R1, R2,
R3

Programmet venter til en key er sluppet før subrutinen vender tilbage med key-
værdien.

Programmet testet - og virker d. 2/11-2016
```

```
 */


const int numRows = 4; // number of rows in the keypad
const int numCols = 3; // number of columns
const int debounceTime = 20; // number of milliseconds for switch to be stable

// keymap defines the character returned when the corresponding key is pressed
const char keymap[numRows][numCols] = {

  {  '1', '2', '3'   },
  {  '4', '5', '6'   },
  {  '7', '8', '9'   },
  {  '*', '0', '#'   }
};

// this array determines the pins used for rows and columns
const int rowPins[numRows] = { 2, 3, 4, 5 };      // Rows 0 through 3
const int colPins[numCols] = { 6, 7, 8 };         // Columns 0 through 2


void setup()
{
  Serial.begin(9600);
  for (int row = 0; row < numRows; row++)
  {
    pinMode(rowPins[row], INPUT); // Set row pins as input

    //   digitalWrite(rowPins[row],HIGH); // turn on Pull-ups
  }
  for (int column = 0; column < numCols; column++)
  {
    pinMode(colPins[column], OUTPUT); // Column pins as outputs
    digitalWrite(colPins[column], LOW); // Make all columns inactive
  }
}


void loop()
{
  char key = getKey();
  if ( key != 0) { // if the character is not 0 then it's a valid key press
    Serial.print("Got key ");
    Serial.println(key);
  }
}



// returns with the key pressed, or 0 if no key is pressed
char getKey()
{
  char key = 0; // 0 indicates no key pressed
  for (int column = 0; column < numCols; column++)
  {
    digitalWrite(colPins[column], HIGH); // Activate the current column.
    for (int row = 0; row < numRows; row++) // Scan all rows for a key press.
    {
      if (digitalRead(rowPins[row]) == HIGH) // Is a key pressed?
```

```
      {
        delay(debounceTime); // debounce
        while (digitalRead(rowPins[row]) == HIGH)
          ; // wait for key to be released
        key = keymap[row][column]; // Remember which key was pressed.
      }
    }
    digitalWrite(colPins[column], LOW); // De-activate the current column.
  }
  return key; // returns the key pressed or 0 if none
}
```

Mine kits kan desværre ikke bruges til Programmer, hvor der inkluderes et keypad-library !!!
Det er fordi der i biblioteket bruges active high inputs.

Men det virker fint hvis man ” bare ” tilslutter en keypad direkte til en Arduino, uden pul-down-modstande !!

4x4 Keypad Kodeeksempel:

```
/*
Keypad sketch
 prints the key pressed on a keypad to the debug window through  the serial port
 Modificeret af Valle

Programmet virker kun til et 4x4 keypad. Testet d. 27/03-2014

 Til Keypads med Pull Down-modstande.

Pins på Keypad er set fra knapsiden, fra venstre:
Kolonne / Column – Række / Row  K0, K1, K2, K3, - R0, R1, R2, R3

Kolonne 0 er den venstreste, Række 0 er den øverste.

 */
const int numRows = 4;     // def. number of rows in the keypad
const int numCols = 4;      // def.  number of columns
const int debounceTime = 20;     // number of milliseconds for switch to be
stable

// keymap defines the character returned when the corresponding key is pressed

const char keymap[numRows][numCols] = {
```

```
    { '1', '2', '3', 'F' },
    { '4', '5', '6', 'E' },
    { '7', '8', '9', 'D' },
    { 'A', '0', 'B', 'C' }
};

// this array determines the pins used for rows and columns
// Pin 9 til venstre pin på keypad, pin 8 til # 2 osv.

const int rowPins[numRows] = { 5, 4, 3, 2 };  // Rows 0 through 3
const int colPins[numCols] = {   9, 8, 7, 6 };     // Columns 0 through 2

void setup()
{
    Serial.begin(9600);
    for (int row = 0; row < numRows; row++)
    {
        pinMode(rowPins[row],INPUT);      // Set row pins as input
        //   digitalWrite(rowPins[row],HIGH);      // turn on Pull-ups
    }
    for (int column = 0; column < numCols; column++)
    {
        pinMode(colPins[column],OUTPUT);        // Set column pins as outputs for
writing
        digitalWrite(colPins[column],LOW);         // Make all columns inactive
    }
}
void loop()
{
    char key = getKey();
    if( key != 0) { // if the character is not 0 then it's a valid key press
        Serial.print("Got key   ");
        Serial.println(key);
    }
}

char getKey()     // returns with the key pressed, or 0 if no key is pressed
{
    char key = 0;       // 0 indicates no key pressed
    for(int column = 0; column < numCols; column++)
    {
        digitalWrite(colPins[column],HIGH);      // Activate the current column.
        for(int row = 0; row < numRows; row++)     // Scan all rows for a key
press.
        {
            if(digitalRead(rowPins[row]) == HIGH)     // Is a key pressed?
            {
                delay(debounceTime);      // debounce
```

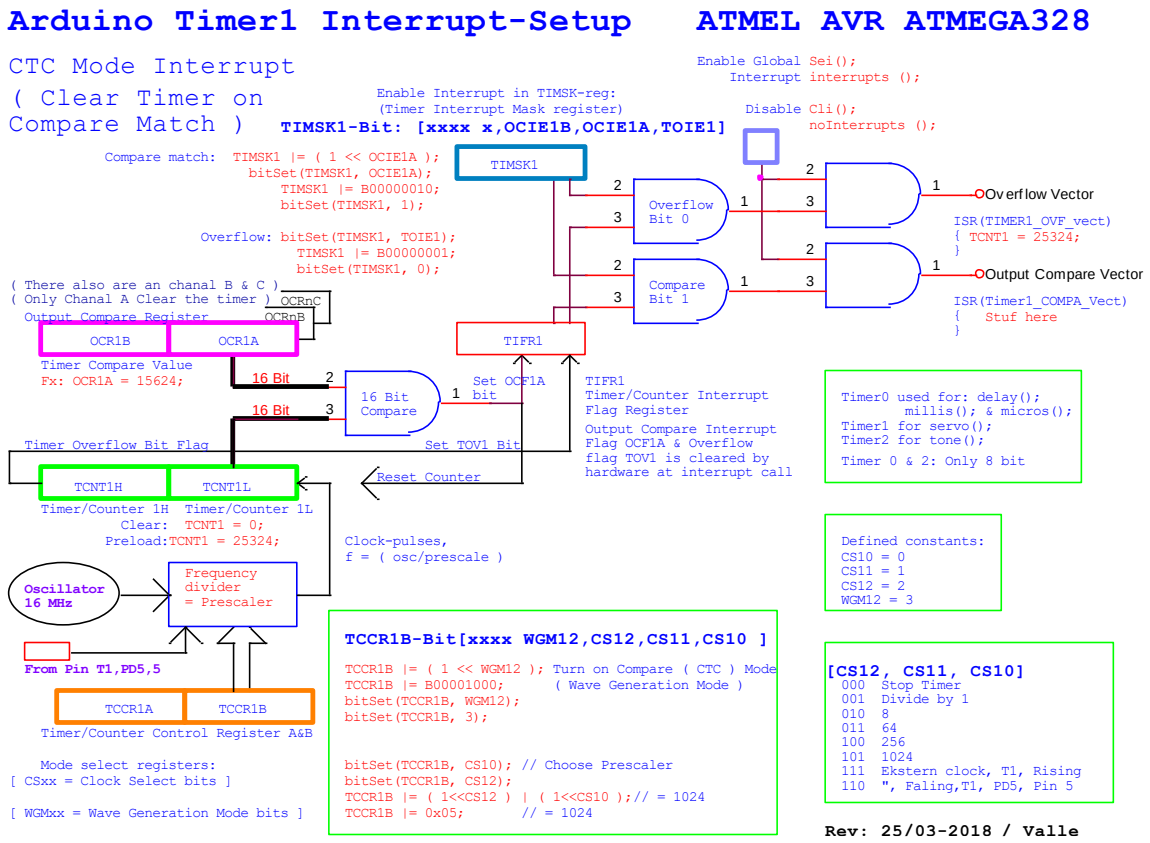
```
                while(digitalRead(rowPins[row]) == HIGH)  ; // wait for key to be
released
                key = keymap[row][column];     // Remember which key was pressed.
            }
        }
        digitalWrite(colPins[column],LOW);      // De-activate the current
column.
    }
    return key;     // returns the key pressed or 0 if none
}
```

Top ↑

Interrupts:

Med følgende diagram har jeg forsøg at lave et samlet diagram over ATMEGA328's timer og interrupt-struktur.

Se for mere uddybende materiale [speciel dokument her:]



I Timer/Counter-registeret opsættes systemet til Compare-mode (ved at sætte bit 3, WGM12 i register TCCR1B) eller Overflow-mode (default ?).

Og der vælges, hvilken frekvensdeler, der ønskes. Det sker med bit 0, 1 og 2, også i register TCCR1B.

Oscillatorens frekvens kommer herefter gennem frekvensdeleren (prescaleren) og ind i en 16 bit tæller.

Prescaleren skal vælges, så tælleren ikke får overflow (> 65.535 pulser) mellem hver interrupt.

I Compare registeret loades det tal, der tælles op til i perioden mellem interrupts.

Når tælleren når op til den værdi, sættes et bit i TIFR1-registeret.

Hvis det tilsvarende bit i TIMSK1-registeret er sat, og den globale interrupt-bit er sat, udløses et interrupt.

Automatisk sker nu, at Loop-programmet afbrydes, Tælleren nulstilles, Bittet i TIFR1-registeret resettes, og programdelen ISR(TIMER1_OVF_vect { } udføres.

Følgende med ??? eller redigeres i hvert fald !!!!!

Eksempel mere:

På Uno-en sidder der et krystal på 16 MHz. Det bruges i næste eksempel:

Der bruges her en frekvensdeler på 1024, dvs. der kommer en frekvens på 16M / 1024 = 15625 Hz til tælleren.

Dvs. når der talt 15.625 pulser, er der gået 1 sekund.

Så der skal indsættes en værdi på 15.624, (fordi tælleren starter med 0) i et sammenligningsregister, og når tælleren kommer op på dette tal, udløses et interrupt, og tælleren nulstilles.

Gennemgå programmet, - og tilføj manglende kommentarer!!

Kodeeksempel:

Følgende program bør omskrives så der bruges bitset i stedet for bitshift ! –

```
/*  Arduino timer/counter Compare Match "CTC" interrupt example
```

```
Urprogrammet er skrevet til 1.z og EUX til at styre tiden i forbindelse med
lysstyring til krydderurter.

Tiden vises på Debug-vinduet.

Der udløses et interrupt hver 1 sekund.

I en interruptrutine optælles sekunder, og der tjekkes for >= 60.
Hvis tilfældet, nulstilles, og minutter øges med 1.
Igen tjekkes for overløb. Osv.

Der er lavet mulighed for at justere uret.

Der er plads til at man selv kan tilrette programmet, så der kan tilføjes
temperaturstyring, - og / eller brug af LCD-display.

Testet 18/3-2015

Valle */


#define LEDPIN 13      // for test

// Definering af Variabler:
byte sekundTaeller =  17;     //  Startværdier, for test
byte minutTaeller =   41;
byte timeTaeller =    17;
byte inByte;


void setup()
{
  pinMode(LEDPIN, OUTPUT);    // for test

  // initialize Timer1 til interrupt
  cli();          // disable global interrupts
  TCCR1A = 0;     // set entire TCCR1A register to 0
  TCCR1B = 0;     // same for TCCR1B

  OCR1A = 15624;    // set compare match register to desired timer count:
  //   Bit i TCCR1B:  **** WGM12, CS12, CS11, CS10
  TCCR1B |= (1 << WGM12);    // turn on CTC mode:

  //   Vælg prescaler og start timer
  //   TCCR1B |= (1 << CS10);                // set prescaler to 1
  //   TCCR1B |= (1 << CS11);                // set prescaler to 8
  //   TCCR1B |= (1 << CS11) | (1 << CS10);  // Set prescaler to 64
  //   TCCR1B |= (1 << CS12);                // set prescaler to 256

  TCCR1B |= (1 << CS12) | (1 << CS10);       // Set prescaler to 1024
  // Eller  TCCR1B |= 0x05;
  TIMSK1 |= (1 << OCIE1A); // enable timer compare interrupt:
  sei();    // enable global interrupts:

  Serial.begin(9600);

}

void loop()
```

```
{
  //  Måske er det bedre at sætte display-håndteringen ned i Interrupt
  //  delen. Så vil den kun skrive på vinduet, når der er gået 1 sekund.

  Serial.print("tiden er: ");
  Serial.print(timeTaeller);

  Serial.print(" : ");
  Serial.print(minutTaeller);

  Serial.print(" : ");
  Serial.println(sekundTaeller);

  if (Serial.available() > 0) {
    inByte = Serial.read();   // get incoming byte:
    if (inByte == 'T') {        // test for Byte
      timeTaeller++;
    }
    else if (inByte == 't') {
      timeTaeller--;
    }
    else if (inByte == 'M') {
      minutTaeller++;
    }

    else if (inByte == 'm') {
      minutTaeller--;
    }
    else if (inByte == 's') {
      sekundTaeller = 0;
    }
  }

  delay(500);
}

ISR(TIMER1_COMPA_vect)  // Interrupt service ( sub )rutine
{
  digitalWrite(LEDPIN, !digitalRead(LEDPIN));  // toggle pin.

  sekundTaeller++;

  if (sekundTaeller >= 60) {
    sekundTaeller = 0;
    minutTaeller++;
  }

  if (minutTaeller >= 60) {
    minutTaeller = 0;
    timeTaeller++;
  }

  if (timeTaeller >= 24) {
    timeTaeller = 0;
  }
}
```

Eksempel på Counter Compare Match interrupt:

```
/*  Arduino timer/counter Compare Match "CTC" interrupt example

  Testet 8/11-2013.  1 sek. Interrupt ???

  Valle */


#define LEDPIN 13

void setup()
{
  pinMode(LEDPIN, OUTPUT);

  // initialize Timer1
  cli();          // disable global interrupts
  TCCR1A = 0;     // set entire TCCR1A register to 0
  TCCR1B = 0;     // same for TCCR1B

  OCR1A = 15624;    // set compare match register to desired timer count:
  bitSet(TCCR1B, 3);  // Bit 3 I Timer control reg. turn on CTC mode:

  //  Vælg prescaler og start timer

  bitSet(TCCR1B, 2);  // Set prescaler to 1024
  bitSet(TCCR1B, 0);  // Set prescaler to 1024


  bitSet(TIMSK1, 1);       // enable timer compare interrupt:

  sei();    // enable global interrupts:


}

void loop()
{
  // do some stuff while my LED keeps blinking
}

ISR(TIMER1_COMPA_vect)
{
  digitalWrite(LEDPIN, !digitalRead(LEDPIN));  // toggle pin.



}
```

Top ↑

Eksempel på Counter overflow interrupt.

```
/*
  Eksempel på Interrupt ved timer overflow.

  Valle / 8/11-2013

*/

// #define ledPin 13, hvis den skal bruges
int timer1_startvalue;
int sekund = 0;
int minut = 0;
int hun_delsekund = 0;
volatile boolean flag = 0;  // global variabel


void setup()
{
  //    pinMode(ledPin, OUTPUT);

  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for Leonardo only
  }


  // initialize timer1
  noInterrupts();           // disable all interrupts
  TCCR1A = 0;
  TCCR1B = 0;

  // Set timer1_startvalue to the correct value for our interrupt interval
  //timer1_startvalue = 64886;   // preload timer 65536-16MHz/256/100Hz
  //timer1_startvalue = 64286;   // preload timer 65536-16MHz/256/50Hz
  //timer1_startvalue = 34286;   // preload timer 65536-16MHz/256/2Hz
  timer1_startvalue = 3036;   // preload timer 65536-16MHz/256/1Hz

  TCNT1 = timer1_startvalue;   // preload timer
  bitSet(TCCR1B, 2);   // set prescaler to 256
  bitSet(TIMSK1, 0);   // enable timer overflow
  interrupts();        // enable all interrupts again !!
}

ISR(TIMER1_OVF_vect)        // interrupt service routine
{
  TCNT1 = timer1_startvalue;   // gen-load timer1

  //     digitalWrite(ledPin, digitalRead(ledPin) ^ 1); Toggle evt. Led

  hun_delsekund++;
```

```
  flag = HIGH;
  if (hun_delsekund > 99) {
    hun_delsekund = 0;
    sekund++;

    if (sekund > 59) {
      sekund = 0;
      minut++;

    }
  }
}

void loop()
{
  while (flag == LOW) { // Wait until change !!
  }
  Serial.print(minut);
  Serial.print(':');
  if (sekund < 10) Serial.print('0');
  Serial.print(sekund);
  Serial.print(':');
  if (hun_delsekund < 10) Serial.print('0');
  Serial.println(hun_delsekund);
  flag = 0;

}
```

Test programmet, - og prøv fx at ændre prescaleren.

Se flere eksempler: http://letsmakerobots.com/node/28278

Eksempel på 2 Hz interrupt der bruger Counter Compare.

```
/* Arduino: timer and interrupts
   Timer1 compare match interrupt example
   more infos: http://www.letmakerobots.com/node/28278
   created by RobotFreak
*/

#define ledPin 13

void setup()
{
  pinMode(ledPin, OUTPUT);

  // initialize timer1
  noInterrupts();           // disable all interrupts
  TCCR1A = 0;
  TCCR1B = 0;
  TCNT1  = 0;

  OCR1A = 31250;            // compare match register 16MHz/256/2Hz
```

```
  bitSet(TCCR1B, 3);   // Turn on CTC mode
  bitSet(TCCR1B, 2);   // 256 prescaler
  bitSet(TIMSK1, 1);   // enable timer compare interrupt
  interrupts();             // enable all interrupts
}

ISR(TIMER1_COMPA_vect)          // timer compare interrupt service routine
{
  digitalWrite(ledPin, digitalRead(ledPin) ^ 1);   // toggle LED pin
}

void loop()
{
  // your program here...
}
```

Timer 1 overflow interrupt eksempel: 2 Hz

```
/*
 * Arduino: timer overflow interrupts
 * Timer1 overflow interrupt example
 * more infos: http://www.letmakerobots.com/node/28278
 *
 */

#define ledPin 13

void setup()
{
  pinMode(ledPin, OUTPUT);

  // initialize timer1
  noInterrupts();           // disable all interrupts
  TCCR1A = 0;
  TCCR1B = 0;

  TCNT1 = 34286;            // preload timer 65536-16MHz/256/2Hz
  TCCR1B |= (1 << CS12);    // 256 prescaler
  TIMSK1 |= (1 << TOIE1);   // enable timer overflow interrupt
  interrupts();             // enable all interrupts
}

ISR(TIMER1_OVF_vect)        // interrupt service routine that wraps a user
      // defined function supplied by attachInterrupt
{
  TCNT1 = 34286;            // preload timer
  digitalWrite(ledPin, digitalRead(ledPin) ^ 1); // toggle LED pin
}

void loop()
{
  // your program here...
}
```

```
// timer example from electronicsblog.net
#define LED 13

boolean x = false;


void setup() {

  pinMode(LED, OUTPUT);

  TIMSK1 = 0x01; // enabled global and timer overflow interrupt;
  TCCR1A = 0x00; // normal operation page 148 (mode0);
  TCNT1 = 0x0BDC; // set initial value to remove time error (16bit counter
register)
  TCCR1B = 0x04; // start timer/ set clock

};

void loop () {

  digitalWrite(LED, x);

};

ISR(TIMER1_OVF_vect) {
  TCNT1 = 0x0BDC; // set initial value to remove time error (16bit counter
register)
  x = !x;
}
```

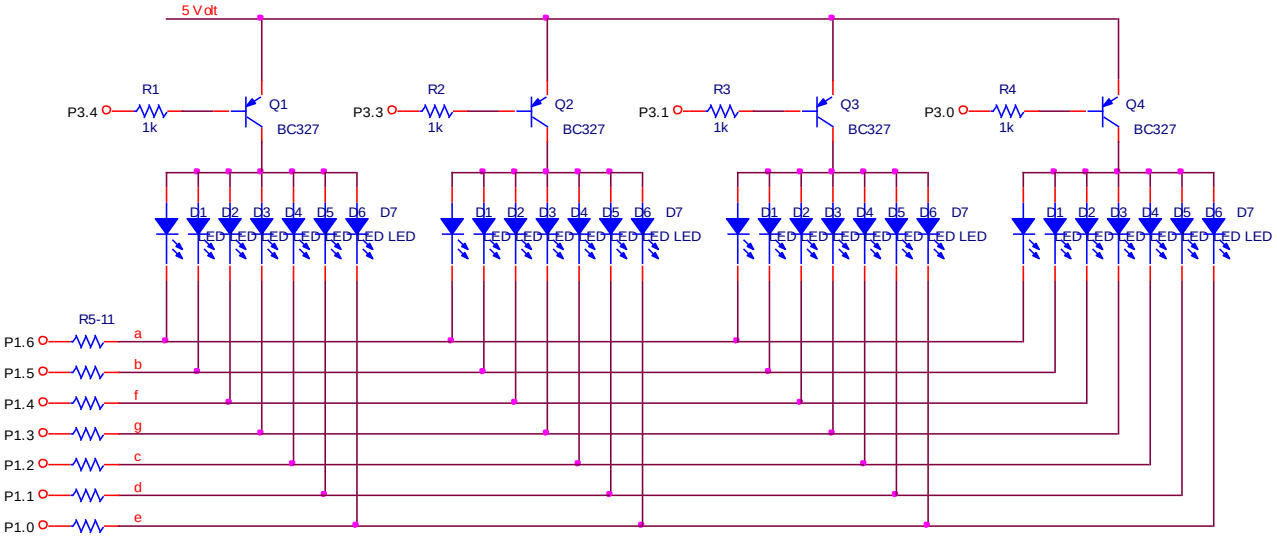
Top ↑

Stopur

Opbyg 4 x 7-segment, med multiplex transistorer.

Fx kan følgende diagram bruges.

Evt. kan der bruges en ULN2003 til at trække strøm fra 7-segmenterne.



Top ↑

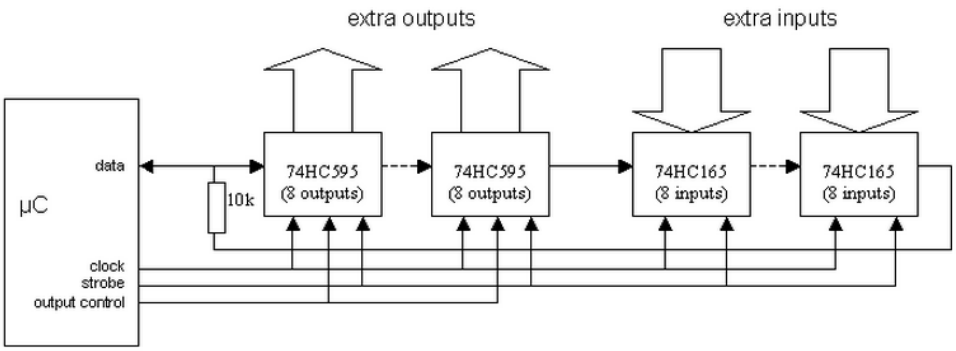
Udvidelse af pinantal:

Default har Arduino ikke mange pins til omverdenen, - men ved at anvende eksterne skifteregistre, kan man let udvide med flere outputs og / eller inputs:

Fx kan man bruge IC-erne 74HC595 til ekstra outputpins, og 74HC165 til flere inputpins.

Og ret smart, er der i IDE'en indbyggede biblioteker der kan håndtere kredsene.

Flere udgange og indgange kan opnås med kredsene 74HC595 og 74HC165



Se: http://robots.freehostia.com/Software/ShiftRegister/ShiftRegister.html

IC-erne kan kaskadekobles, (forbindes i serie) – så man med 2 IC-er fx kan få 16 ekstra outputs.

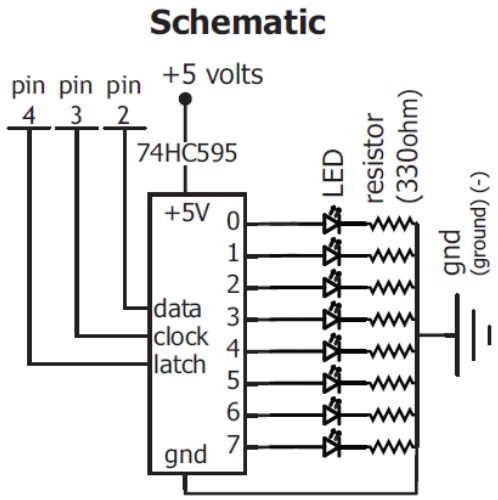
Flere outputs med 74HC595

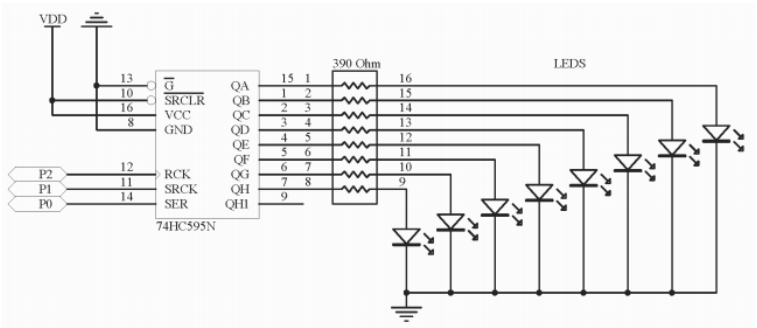
Opbyg og test udvidelse af uC'en med et skifteregister.

Hvis der ikke er nok pins på processoren, er det muligt, vha. et eller flere skifteregistre, at tilføje flere pins.

Med en 74HC595 kan der tilføjes flere outputs, med en 74HC165 flere inputs.

http://arduino.cc/en/tutorial/ShiftOut





Se evt. youtube: http://www.youtube.com/watch?v=bqfPZXEuyuc

Se http://ardx.org/src/circ/CIRC05-code.txt
http://arduino.cc/en/Tutorial/ShiftOut

Her følger et par program-eksempler:

```
//Pin Definitions
//The 74HC595 uses a serial communication
//link which has three pins

int data = 2;
int clock = 3;
int latch = 4;

void setup()                    //runs once
{
pinMode(data, OUTPUT);
pinMode(clock, OUTPUT);
pinMode(latch, OUTPUT); }

void loop()                     // run over and over again
{
```

```
int delayTime = 100;            //delay between LED updates
for(int i = 0; i < 256; i++){
updateLEDs(i);
delay(delayTime); }
}

/*  updateLEDs() - sends the LED states set
* in value to the 74HC595 sequence
*/

void updateLEDs(int value){
digitalWrite(latch, LOW);               //Pulls the chips latch low
shiftOut(data, clock, MSBFIRST, value);         //Shifts out 8 bits to the shift
register
digitalWrite(latch, HIGH);              //Pulls the latch high displaying the
data
}
```

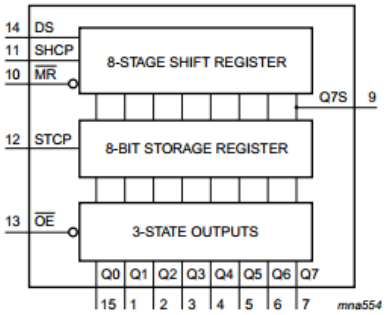
Se: http://arduino.cc/en/Reference/shiftOut

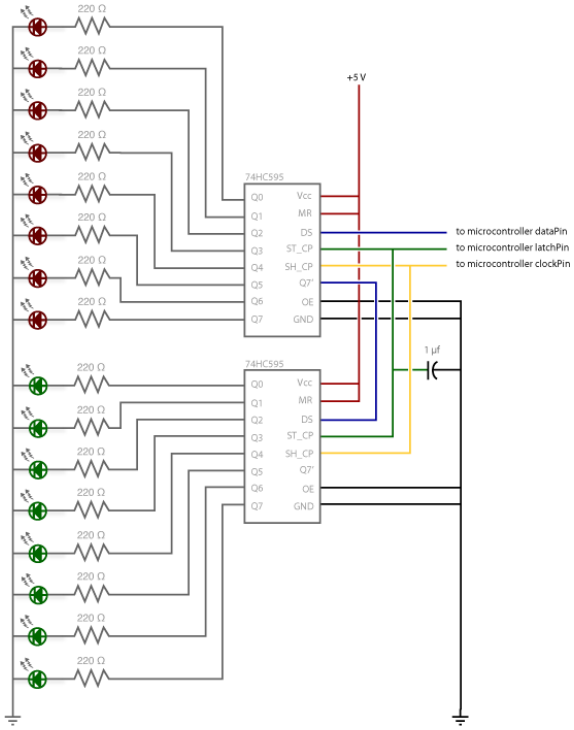
```
//**************************************************************//
//  Name    : shiftOutCode, Hello World                         //
//  Author  : Carlyn Maw,Tom Igoe                               //
//  Date    : 25 Oct, 2006                                      //
//  Version : 1.0                                               //
//  Notes   : Code for using a 74HC595 Shift Register           //
//          : to count from 0 to 255                            //
//**************************************************************


int latchPin = 8;       //Pin connected to ST_CP of 74HC595
int clockPin = 12;      //Pin connected to SH_CP of 74HC595
int dataPin = 11;       //Pin connected to DS of 74HC595

void setup() {

  pinMode(latchPin, OUTPUT);
  pinMode(clockPin, OUTPUT);
  pinMode(dataPin, OUTPUT);
}

void loop() {

  for (int j = 0; j < 256; j++) {               //count up routine

    digitalWrite(latchPin, LOW);        //ground latchPin and hold low for as
long as transmitting
    shiftOut(dataPin, clockPin, LSBFIRST, j);           //  Transmit
    digitalWrite(latchPin, HIGH);       //return the latch pin high
```

```
    delay(1000);
  }
}
```

Kaskadekobling af skifteregistere, 74HC595
Her følger et antal eksempler med diagrammer.

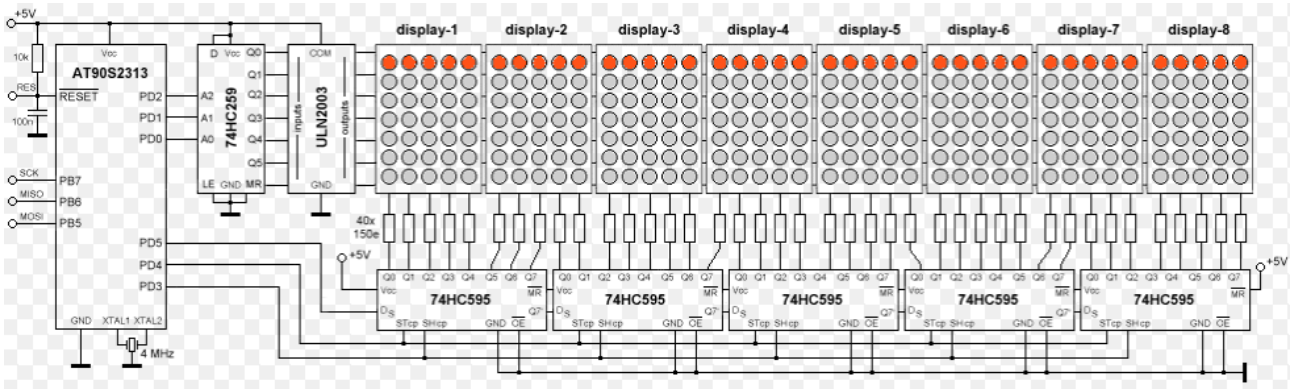
http://arduino.cc/en/tutorial/ShiftOut

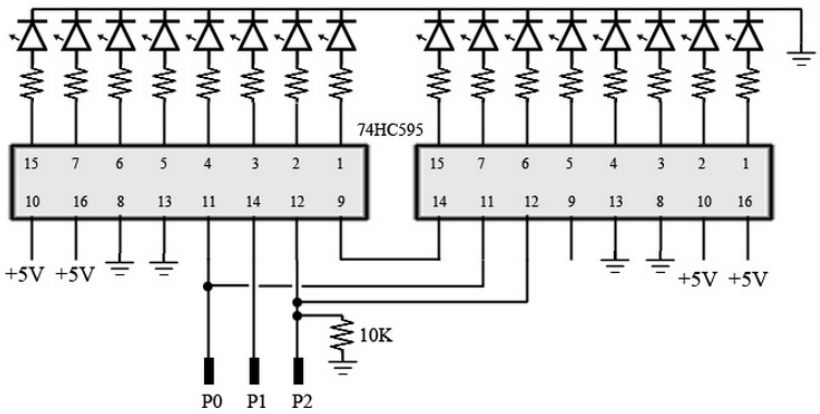


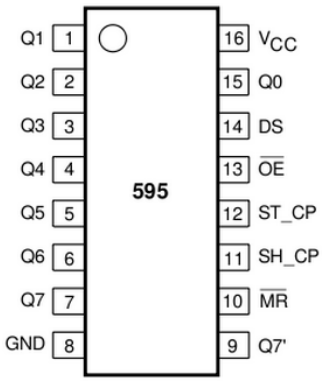


Se: http://arduino.cc/en/Tutorial/ShiftOut

Eksempel:



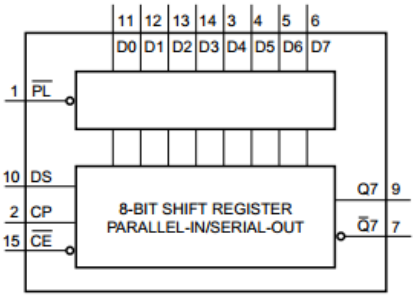


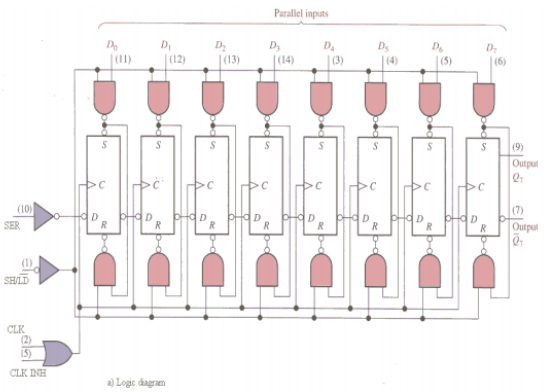


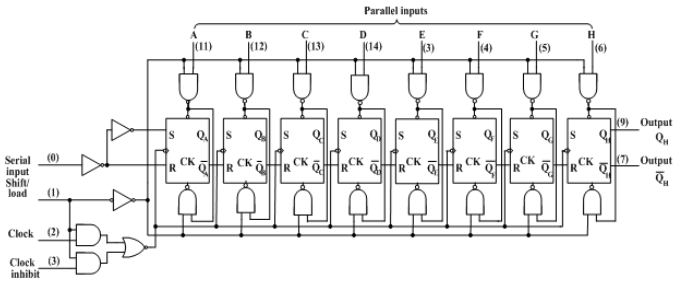
Top ↑

Flere input med 74HC165

74HCT165 are 8-bit parallel-load or serial-in shift registers



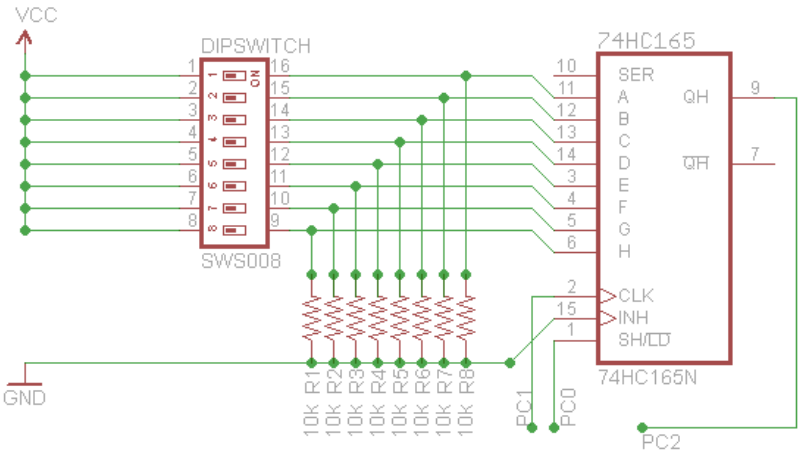




https://maxwell.ict.griffith.edu.au/yg/teaching/dns/dns_module3_p3.pdf

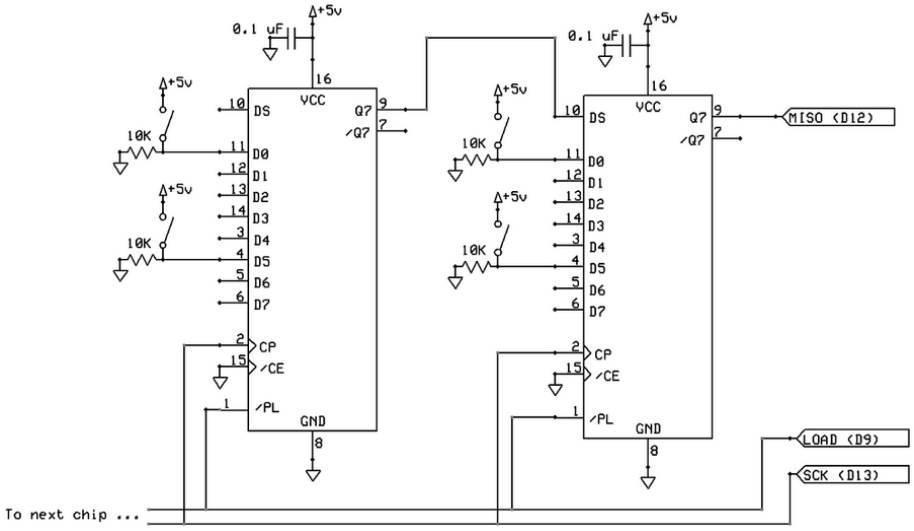
Se eksempel: http://www.gammon.com.au/forum/?id=11979
Se: https://maxwell.ict.griffith.edu.au/yg/teaching/dns/dns_module3_p3.pdf

Her er vist et eksempel på, hvad man kan bruge en skifteregistere-udvidelse af en uC til.

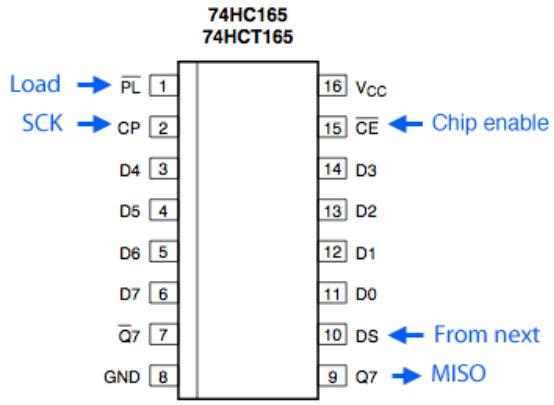


Kilde: http://www.nerdkits.com/forum/thread/1867/

Kaskadekobling af to 8-bit skifteregistere

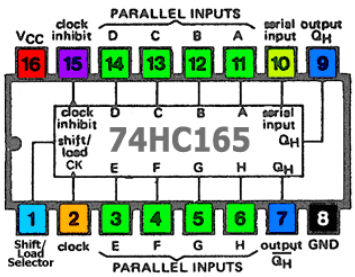


http://www.gammon.com.au/forum/?id=11979



MISO star for Master In, Slave Out.

Det er den pin, der forbindes til Data in på uC'en.



Top ↑

Intern EEPROM

Lav et program, der hvert minut gemmer gennemsnittet af 10 temperaturmålinger i den interne EEPROM
Der skal bruges en LM35 til temperaturmåling, og der skal bruges en intern 1,1 V reference!

Lav dernæst et andet program, der kan hente data ud igen og fx vise dem i debugvinduet eller LCD

Her lidt inspiration:

```
// EEPROM = 1024 Byte


#include <EEPROM.h>
BYTE value;
BYTE address;
address = 0;
EEPROM.write(address++, 0x50);
EEPROM.write(address++, 0x51);

address = 0;
value = EEPROM.read(address++);
value = EEPROM.read(address++);
```

Top ↑

Extern EEPROM

Lav et program der kan måle fx en temperatur via en LM35.

Gem temperaturen i EEPROM'en.

Tag nu strømmen af.

Og senere power on igen, - og lav et program, der kan hente data og vise dem på Debugskærmen eller LCD.

Se fx denne hjemmeside:
https://www.norwegiancreations.com/2017/02/using-eeprom-to-store-data-on-the-arduino/

Top ↑

Små Kit-opgaver:

Brug 1 af de små kits til at forbinde Arduinoen til omverdenen:

MOSFET Relæ. Bruges til at tænde 12 Volts belastninger, fx en motor eller akvariepumpe.

H-Bro kit til at styre 12 Volt DC motorer – så de kan køre højre eller venstre rundt, eller stoppe.

Solid State Relæ: Bruges til at tænde 230 Volts belastninger, - pærer, blæser, varmeovn mm.

Top ↑

Opgaver, der mangler:

HBro, Bipolar stepmotor, brug L293 Dil-kreds, eller L298, Se side 318.

I2C, CB side 421
SPI, CB side 424

Top ↑

Kopier til word med farver

Når kode skal kopieres til en rapport i Word, ser det godt ud, at kodens syntax er farvelagt.

Det kan ske med hjælp fra en række hjemmesider. Google fx ”colorize code” eller ”Source Code beautifier”

Eller se på min hjemmeside her: http://vthoroe.dk/Elektronik/Arduino/Tips_og%20Trix.pdf