Arduino opgaver Teknologi:

Find Hjælp:

Nettet, søg "Arduino" + et søgeord mere
Evt. youtube, - en god kilde er https://www.jeremyblum.com/category/arduino-tutorials/

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

I dette dokument er der inspiration til en række programmerings-opgaver til Arduino.

Opgaverne bliver i nogen grad sværere og sværere.

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

På Kittene er der et 4 linierx20 karakter LCD-display, 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 skal blot have konstant 5 Volt hvorefter den giver lyd på ca. 2 kHz. Den anden skal have tilført den frekvens, der skal høres.

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 så at modificere programmerne.

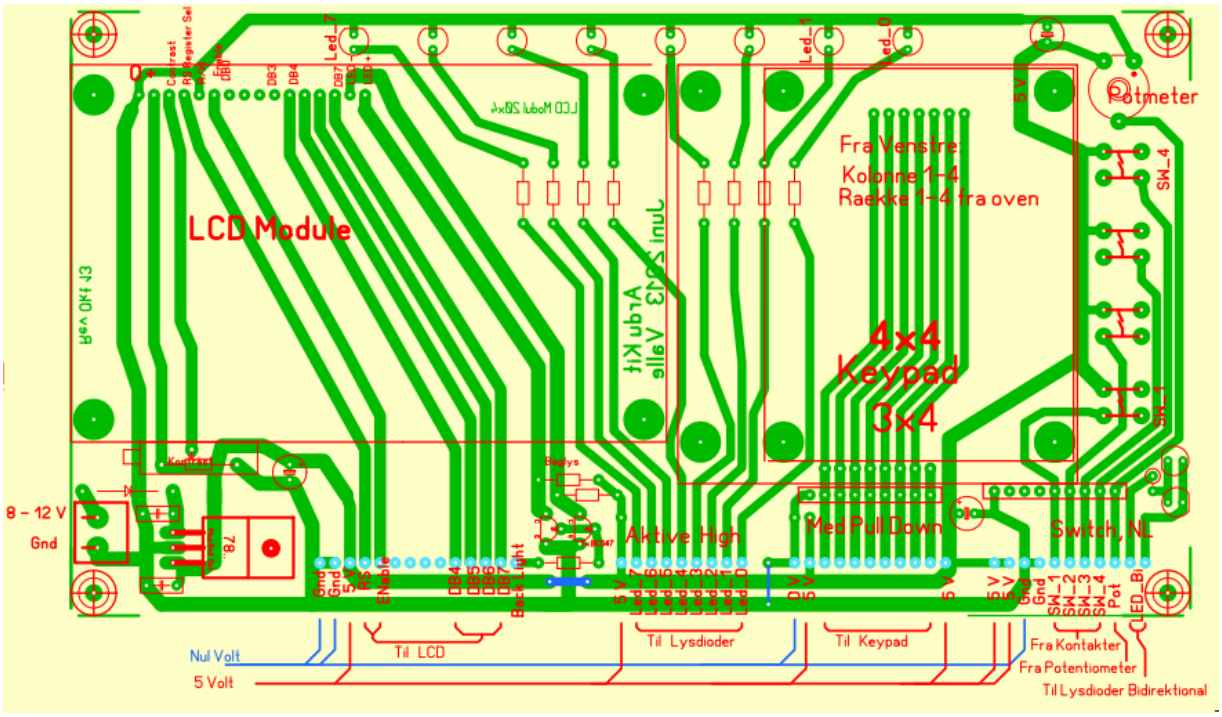
Til de fleste opgaver er der et eksempel, der kan C&P ind i Arduino IDE'en og afprøves. Men ideen er jo, at man selv skal bearbejde, ændre i programmet eller tilføje mere funktionalitet.

Mine Arduino-kit:

For at kunne forbinde Kittene til Arduinoen, er der her gengivet kopier af deres printudlæg.

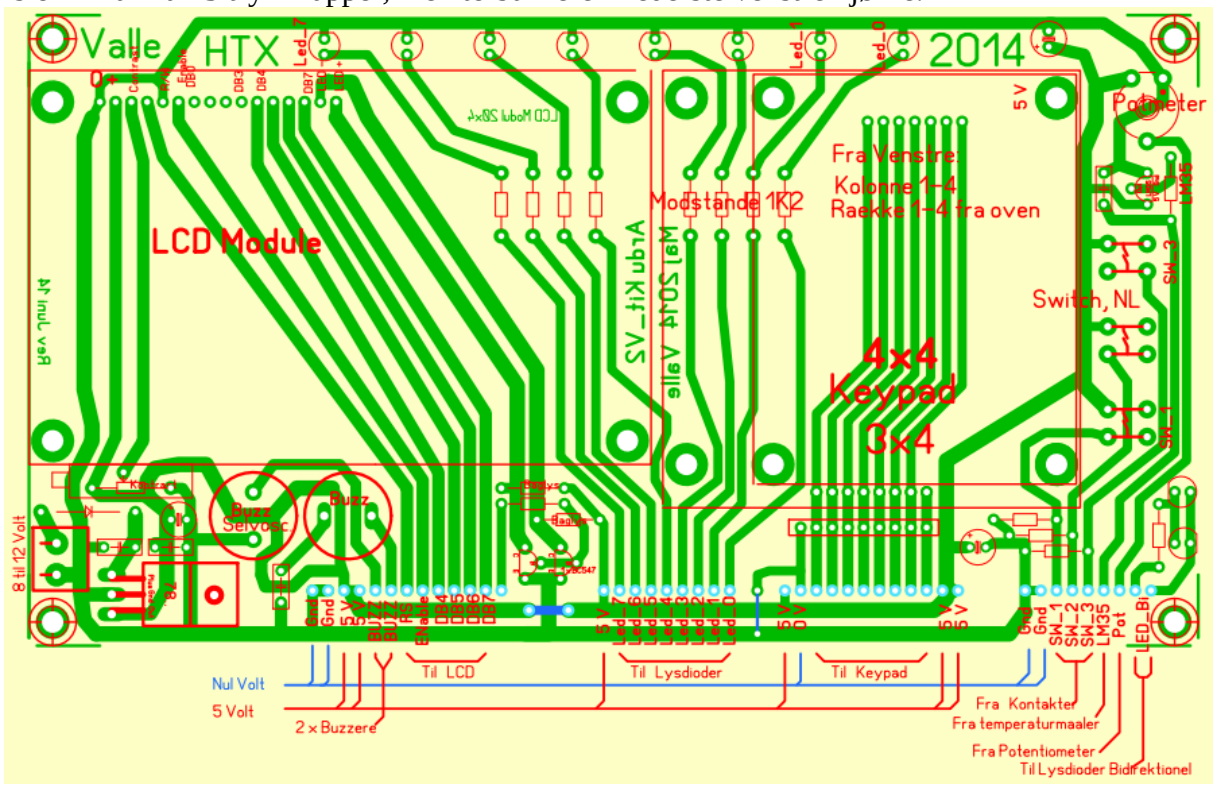
Bemærk, at der er to versioner. De kan fx kendes på antal trykknapper i højre side.

Version 1:



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.

For evt. at finde hjælp til at lave opgaverne kan man søge på nettet. Der findes der et hav af eksempler.

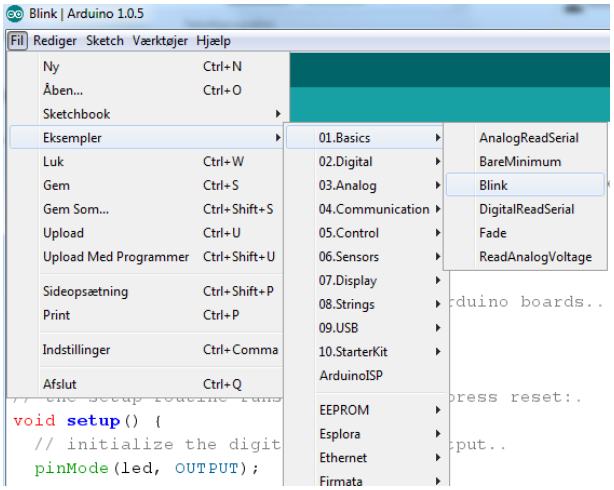
Første øvelse:

Hent sketchen ”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 syd for Sønderborg.

Blinkende Lysdioder

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.

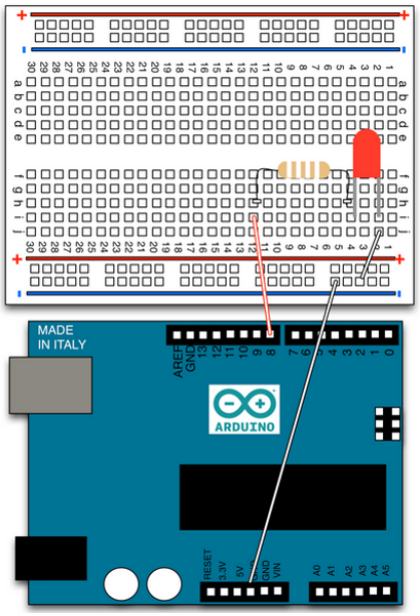
Husk at definere alle pins som outputs i setup!

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.



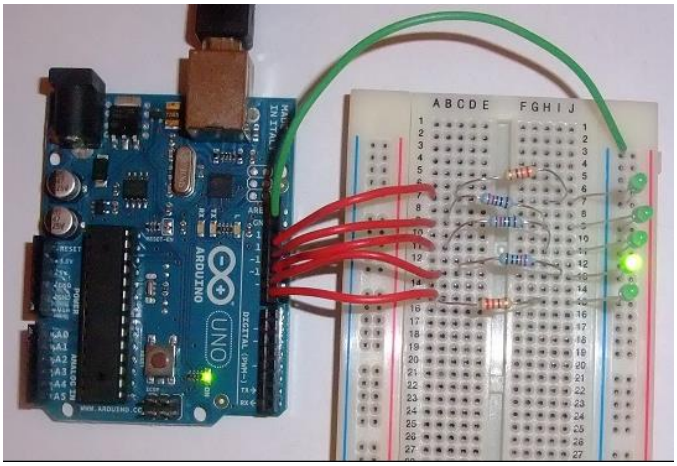
Her et Fumlebrædt set forfra.

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,



Input fra knap

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

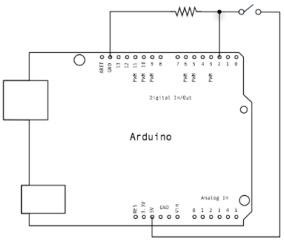
Her er der behov for, at programmet kan læse en kontakt, forbundet til en input-pin.

Her et eksempel på forbindelser.

Bemærk, at der er trykknapper på mine kits.

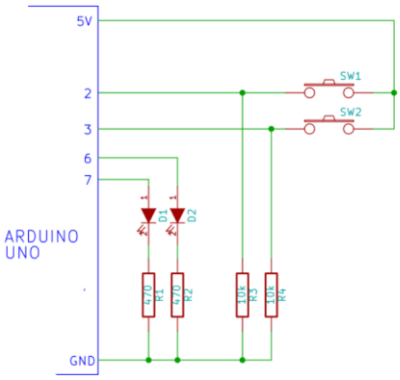
Kittet skal have 5 Volt, 0 Volt, og en forbindelse fra en pushbutton.

Når der trykkes på knappen, bliver signalet højt.



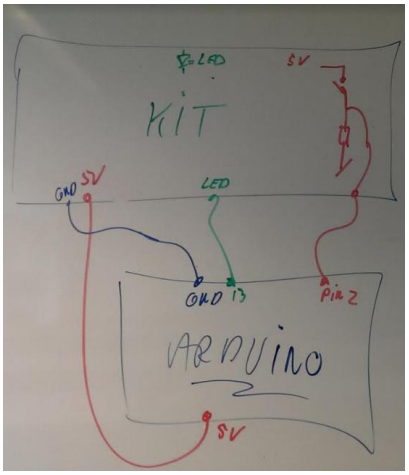
Når der trykkes på en knap, kan der løbe strøm ned gennem modstanden. Herved bliver spændingen over modstanden – og dermed også på pin-en høj. (5 Volt).

Hvis modstanden undlades, vil ledningen hen til pin 2 og 3 svæve. Dvs. ikke være forbundet til noget. Dvs. der meget let dannes 50 Hz signal i ledningen. Det sker på grund af elektriske og magnetiske felter fra vores elnet.



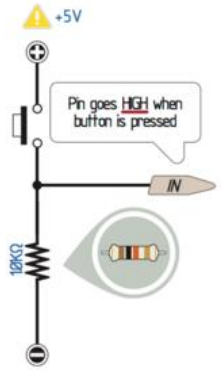
Knapper, - eller buttons – eller switch-es fås i forskellige udformninger. Til forskellige formål.

Forbindelser mellem Unoen og mine kits kan laves sådan!



Nu skal der bruges en program-funktion der kan læse en pin. 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 !! den får programmet til at teste om en betingelse er opfyldt. Og hvis, udføres nogle programlinjer.

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) {

// Do something
    }
}
```

Find koden fx 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 på: http://vthoroe.dk/Teknologi/Arduino/Loops%20mm.pdf

```
void loop()
{

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

Brug af – If – Else Eksempel:

```
//  If - Else:   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, - eller værdien 1.

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

Variabel Blinkfrekvens

Nu skal der laves 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
```

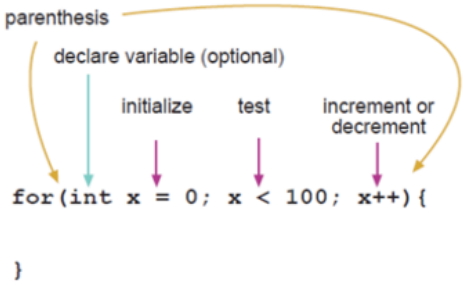
```
byte buttonState = 0;     // skal bruges til at læse værdien af en knap,
                                om den er lav eller høj.


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.

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 have værdien 0, og koden mellem { og } udføres første gang.

Herefter testes om x i dette tilfælde er < 100.

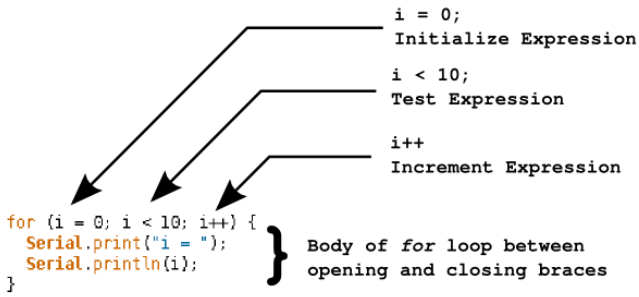
Hvis sand øges x med 1.

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

Herefter gentages koden mellem { og }.

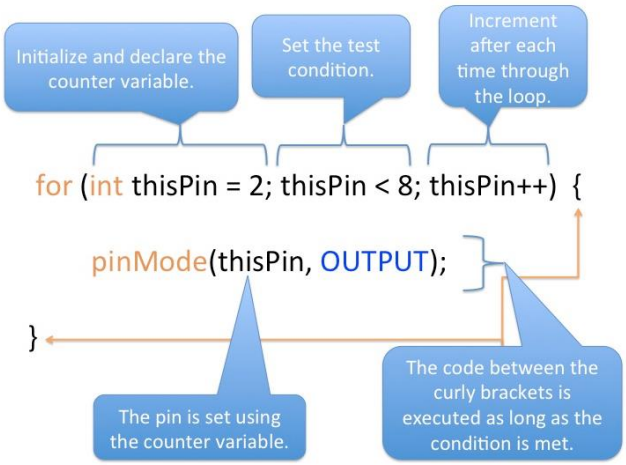
Værdien af variablen x kan så også bruges i koden mellem { og }.

Eksempel:



Her en anden forklaring !!

Her et andet eksempel på en for-loop. Den gør 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);
  }
}
```

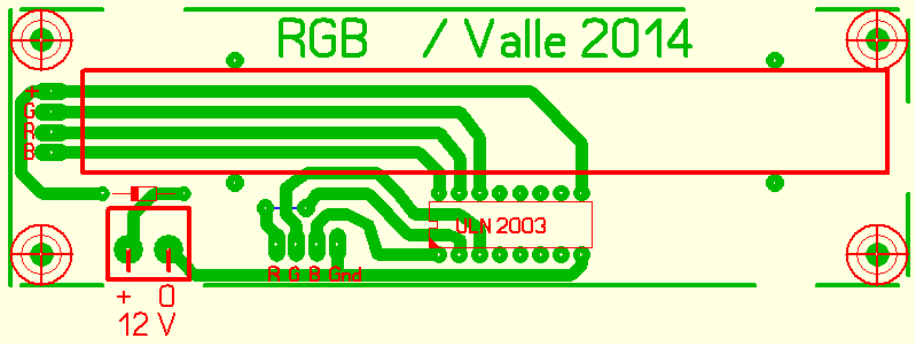
RGB-Kit:

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

Øjet 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 en rød, grøn og blå lysdiode vil hjernen opfatte lyset som en anden farve.

RGB-kittet ser således ud.

Det skal have 12 Volt forsyning fra fx en Netadapter.



På kittet sideer 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.

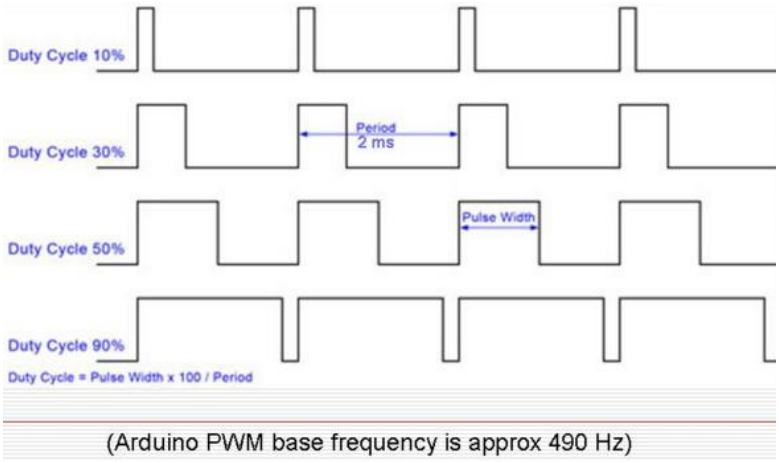
Men selvfølgelig vil lysmængden dioden udsender blive begrænset, hvis dioden kun er tændt halvdelen af tiden. Men hvis lysdioden fx er tændt i ¾ af tiden, vil den lyse kraftigere.

Begrebet kaldes ”PulsBredde-Modulation”, forkortet PWM.

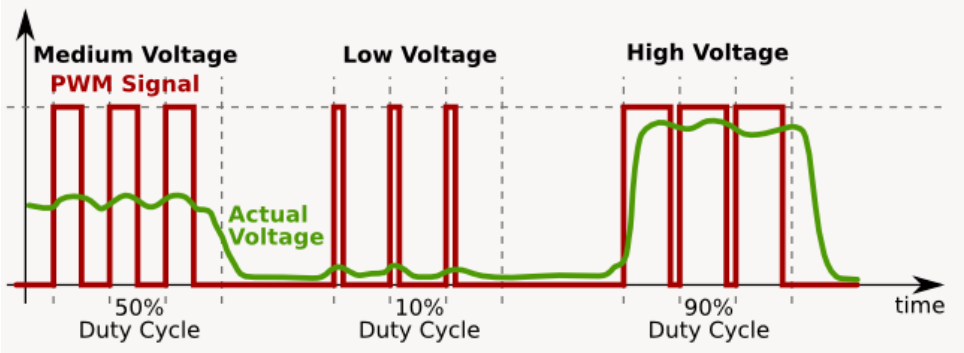
Og begrebet ”Duty Cycle” beskriver i hvor mange procent af en periodetid, et signal er høj.

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 nu skal man i stedet for digitalWrite bruge analogWrite.

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

Hvis R, G & B på kittet forbindes direkte til PWM-outputtene, kan man nu dæmpe, eller fade lyset.

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

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);
   }
}
```

Fadning eksempel:

```
// 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);
   }
}
```

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);
  }
}
```

Her er et eksempel, hvor der er brugt en funktion, der kaldes med 3 parametre.

```
/*
Adafruit Arduino - Lesson 3. RGB LED
*/

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
  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);
}
```

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);
```

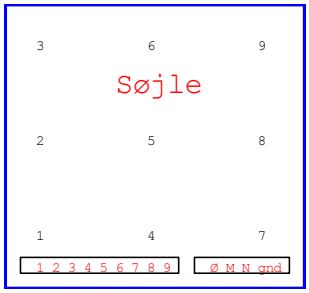
```
      delay(2);
    }
    // pause between LEDs:
    delay(100);
  }
}
```

LED_Cube

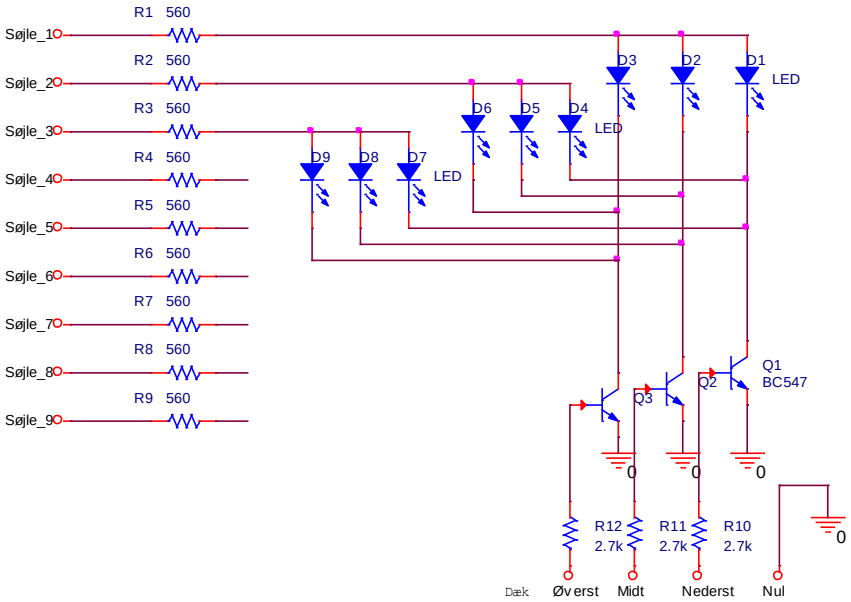
Se fx følgende video: https://www.instructables.com/id/Arduino-Based-3x3-LED-Cube/

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

Diagrammet for dem ser således ud!



Tallene viser søjlenumrene!



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
}
```

LCD .

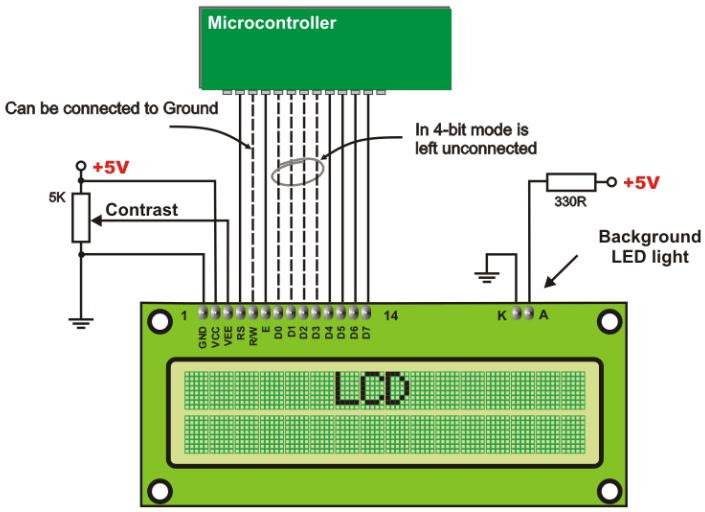
På mine kits er der monteret et LCD-display. Der er lavet de nødvendige forbindelser, der skal til for at få det til at køre.

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

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

Men hvis man selv opbygger et kredsløb med en LCD-skærm er det nødvendigt at montere et potentiometer.

Pin 5 er forbundet direkte til Gnd og de stiplede ledninger D0 til D3 er udeladt.

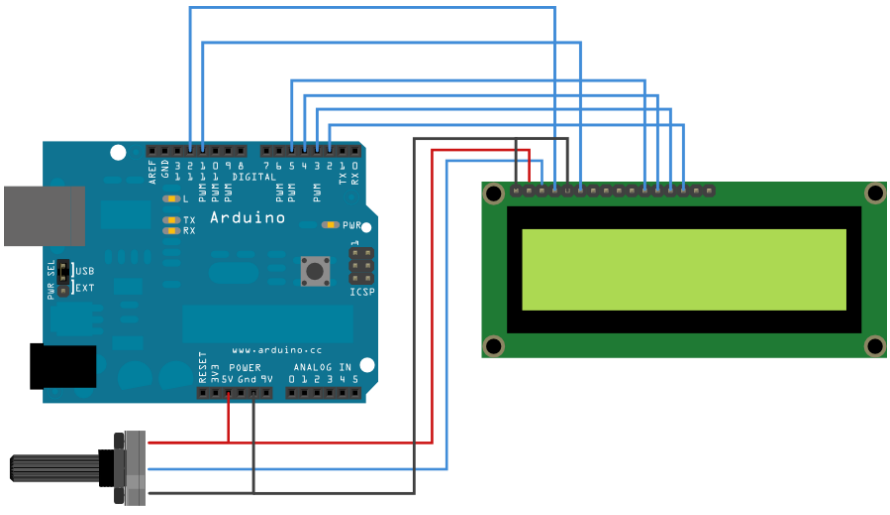


På kittene er der lavet mulighed for at tænde og slukke Backlight.

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

Der findes vist et include-bibliotek til danske karakterer i Arduino-IDE-en.

Ellers se kode på min hjemmeside, under 3. del elektronik / Arduino:

--0--

Start nu med at åbne eksemplet: Fil > Eksempler > Liquid Christal > ”Hello World”

Her et uddrag:

```
// 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!");
}
```

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.

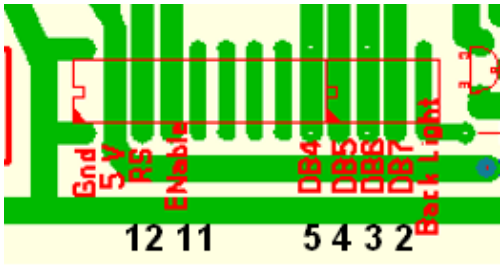
```
    lcd.begin(20, 4);
```

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

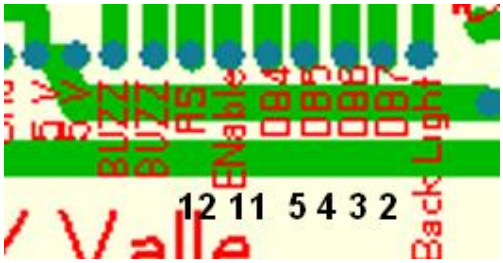
```
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
```

Disse pins fra Unoen skal forbindes til kittets stik som vist her, men bemærk de to versioner:

Version 1 Version 2

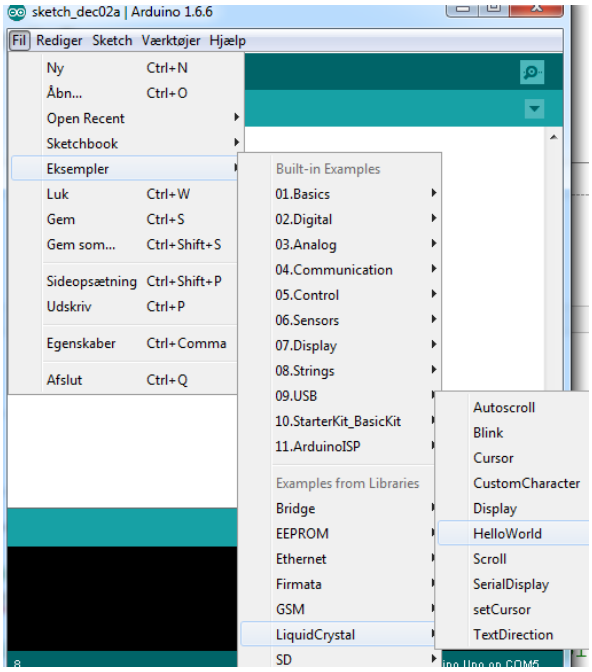


RS skal til pin Arduino pin 12, Enable til pin 11, osv.



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

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);
}
```

Der skal bruges mange nye funktioner, når man vil arbejde med et LCD. De er smart nok lagt ned i et bibliotek. Dette skal inkluderes i koden.

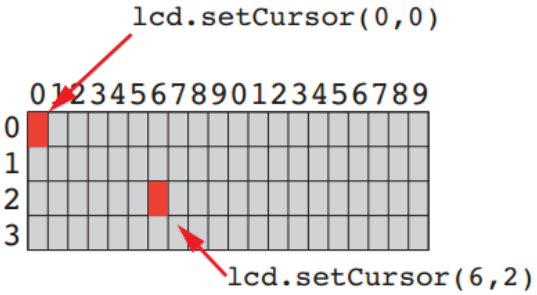
Arbejd nu lidt med koden. Lav om på teksten, der skerives i skærmen.

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

Brug fx `lcd.clear();` `// Clearer alle 4 linjer`

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

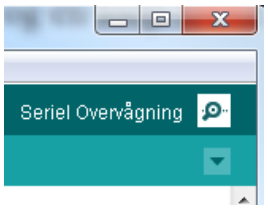
```
lcd.setCursor(x,y);
```



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

Brug debugvinduet

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

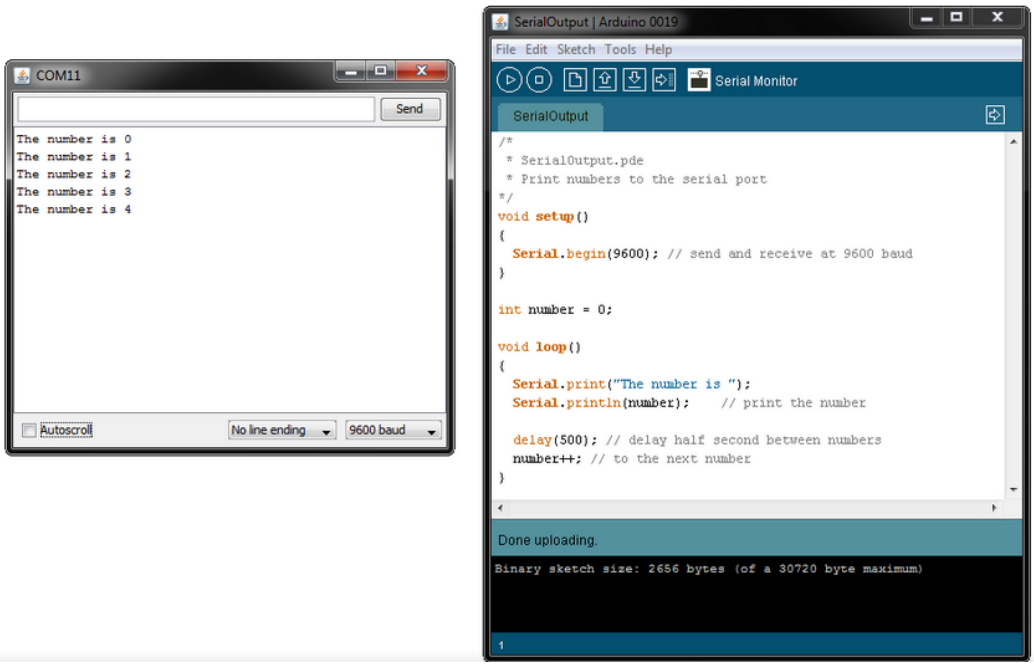


Arduinoen programmeres via kablet, men ud over dette er der mulighed for sende data fra et program i Arduinoen at sende data tilbage til PC-en, men også fra PC-en (Keyboardet) til det program, der kører i uC-en.

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

Vinduet kaldes et Debug-vindue, da det er meget let at afluse – dvs. fejlfinde et program ved fx fra uC-programmet at sende variables værdier til debugvinduet.

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



Bemærk at kommunikationen mellem Boardet og PC-en også bruger pin 0 og 1. De kan derfor ikke bruges samtidig med debug-monitoren til fx at montere lysdioder.

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

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

Eksempel:

```
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()
```

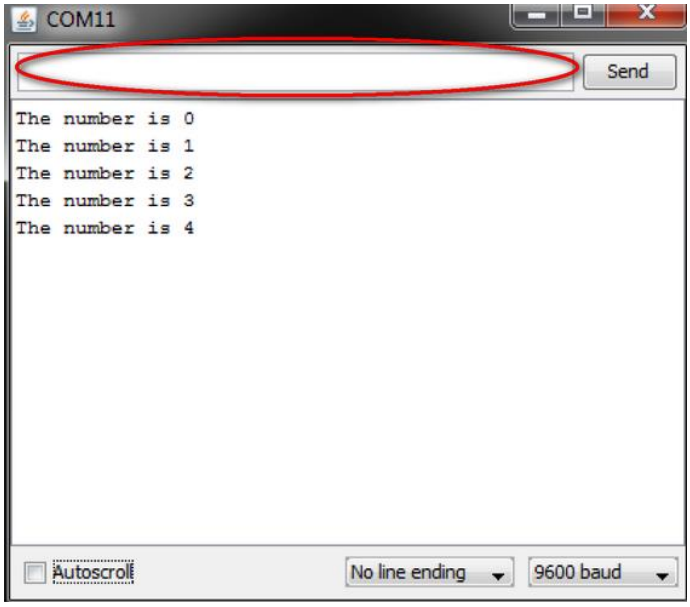
```
{
  Serial.println(x);  // Send værdien af x
  delay(500);
  x=x+1;              // Kan også skrives: x++;
  if (x>5) {x=0;};
}
```

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

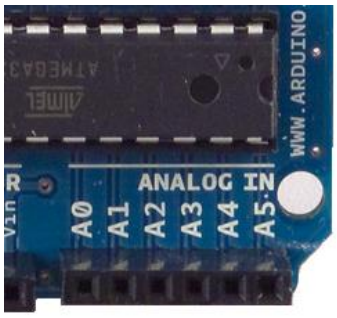
For mere: Se speciel kompendium: http://vthoroe.dk/Elektronik/Arduino/Debugvinduet.pdf

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

Lysdiode-lysstyrke styret af et potentiometer

Mål analog spænding

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



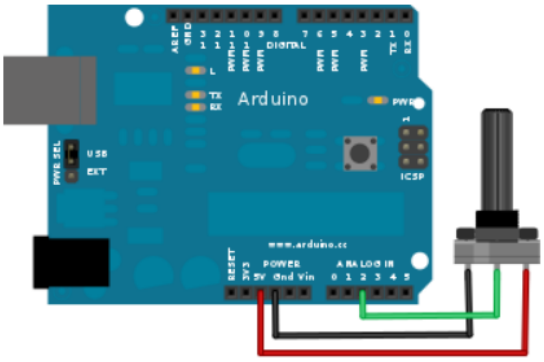
Ordren til at indlæse en værdi er

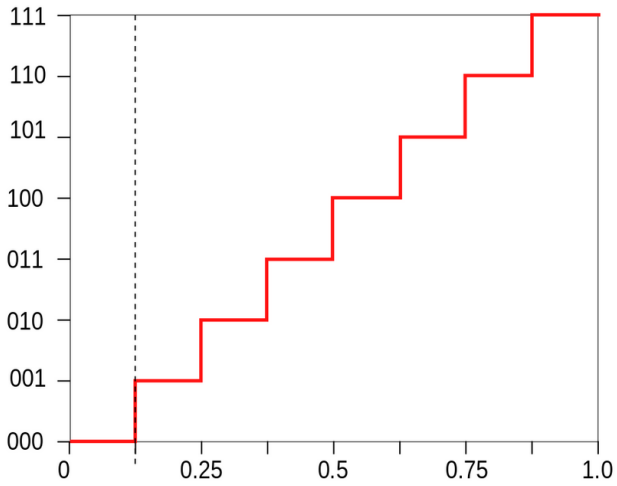
```
Variabel = analogRead(analogIndgang);

Value = analogRead(A0);
```

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

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

Det kan fx bruges til at læse en temperatur fra en temperaturføler.





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 2,5 Volt bliver 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. pauselængden.

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);
// Indsæt din beregning/konvertering her!
  volt = float(analogVaerdi);
  volt = volt / gain;
  Serial.print("\t Spaending = " );
  Serial.println(volt);
  delay(dTime);
}
```

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 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, hvis tallet er et kommatal.

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 #####
```

Keypad

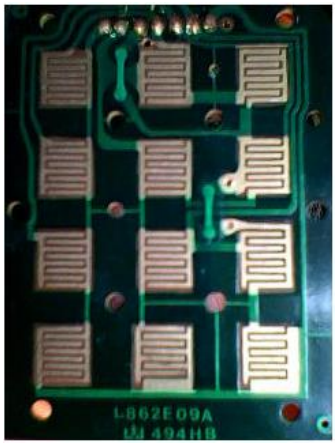
Forbind et Keypad til Arduinoen, - og skriv fx i debugvinduet på PC-en eller en LCD-skærmen, hvilken tast, der trykkes.

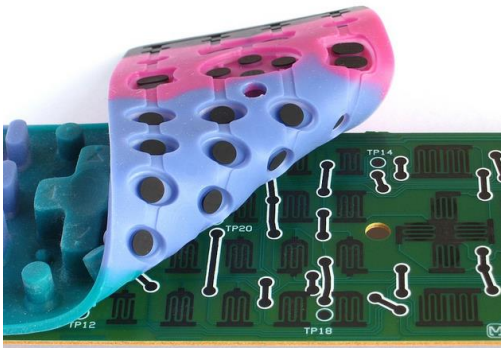
Hvordan virker et Keypad?

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.

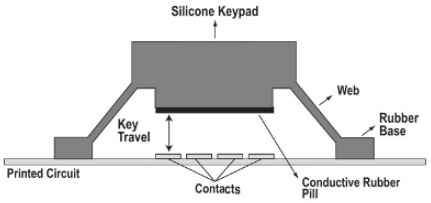
Et keypad kan typisk være konstrueret som vis på disse billeder.

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



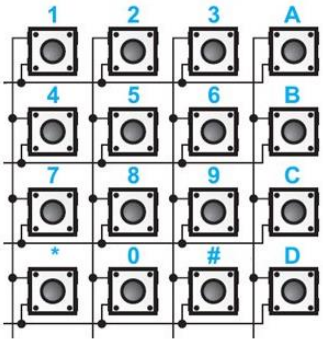


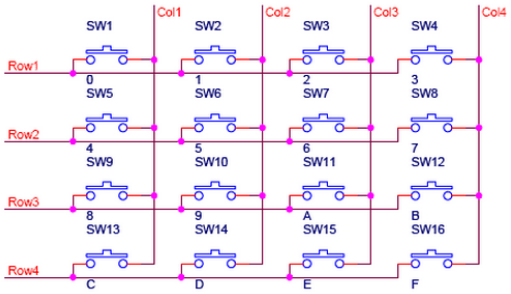
Her er en kontakt vist i skitse



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.





Vore keypads er fra Farnell, Varenummer 1130805 (12 keys) eller 1130806 (16 keys).

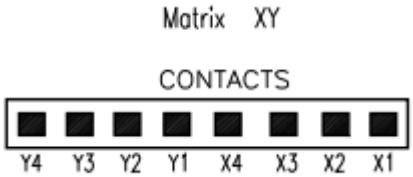
Farnell Varenummer: 1130805

Kolonner og rækker nummereres sådan:

Venstre kolonne er kolonne K1. - Øverste række er R1.

Pins på 4x4 Keypad set forfra er fra venstre:

K1, K2, K3, K4, R1, R2, R3, R4



(Obs: set fra bagsiden !)

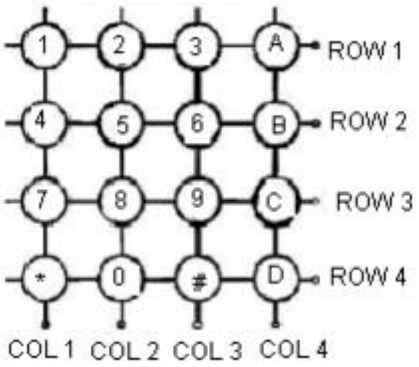
	X1	X2	X3	X4
Y1	1	2	3	F
Y2	4	5	6	E
Y3	7	8	9	D
Y4	A	0	B	C

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);

}
```

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 !!

Termoprinter:

Lav et program, der - når der fx trykkes på en knap - sender en tekst til termoprinteren.

Driveren i termoprinteren se skrevet sådan, at det skal have sendt serielle data med 1200 bit i sekundet, - 1200 Baud.

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

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

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

Herefter skriver printeren en linje.

```
/*
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
{
}
```

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. (For 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>

SoftwareSerial mySerial(10, 11);    // Rx, Tx er valgt til ben 10 og 11
        // I stedet for mySerial kan objektet fx kaldes ” hejMor”
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);   // er jo Input default – vist nok.
  pinMode(tx,OUTPUT);
  digitalWrite(tx,HIGH);
delay(500);

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


  mySerial.begin(1200); // indstil baudrate for SoftwareSerial port
  delay(500);
  mySerial.write( 0x81 );       // ID for første segment
  delay(100);
  mySerial.write ("H");     // Skriv "H"
  delay(100);
 Serial.println("Sendt");       // i debugvinduet
}

void loop() // run over and over
{

}
```

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

Hvis bit 8 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!
```

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.

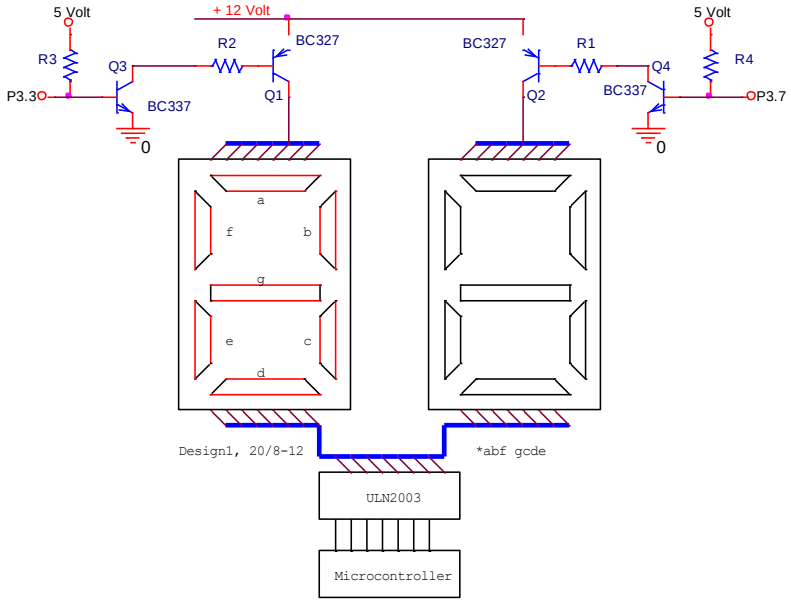
Kittes ID er 8Bh

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 = 8B 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);
    }
  }

}
```

RF-ID

RF-ID står for Radio Frequencies IDentification.

Teknologien kendes fx fra et medarbejderkort der blot skal holdes hen til en læser, for at kortet kan identificeres. På skolen kan kortet give adgang til bygningen uden for normal åbningstid, eller fx bruges til at logge ind på printerne.

Når et kort eller en Tag kommer i nærheden af RF-ID-læseren, sender læseren 10 byte, dvs. 10 digit via dens UART til Arduinoens Rxd.

Først sendes en start-karakter, dernæst 10 digit TAG-nummer, to tjek-cifre og endelig en end-karakter.

Læs nærmere om hvordan systemet virker i et særskilt kompendium her!

Se speciel RF-ID kompendium !!!!!

Se fx: RF-ID: Se: http://tronixstuff.wordpress.com/2010/08/18/moving-forward-with-arduino-chapter-15-rfid-introduction/

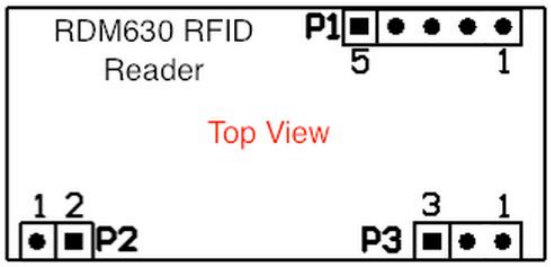
Når en tag kommer i nærheden af læseren, sender læseren kortets nummer i form af 10 byte til Arduinoens RxD.

Først sendes en start-karakter, dernæst 10 digit TAG-nummer, og endelig en end-karakter. (måske er det flere bytes der sendes?)

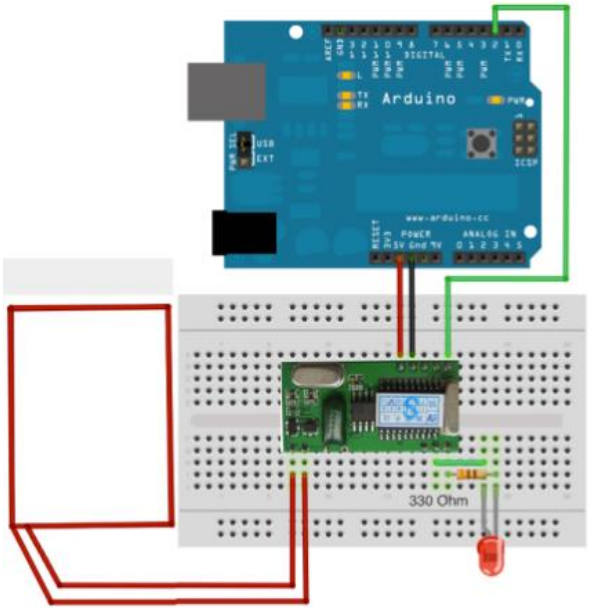
Obs: der kan være forskel på benforbindelserne på de to modeller vi har af RDM630.

Læseren kan læse kortet vha. en spole.

Forbindelserne til model RDM630: New!



P1	P2	P3
PIN1---TX	PIN1---ATN1	PIN1---LED
PIN2---RX	PIN2---ATN2	PIN2---+5v(DC)
PIN3		PIN3---GND
PIN4---GND		
PIN5---+5v(DC)		



Interface Data output format for RF-ID læserne:

Her lidt teknisk information om data fra læseren til uC-en.

1. 9600bps,N,8,1
2. CHECKSUM: card 10byte DATA entire do XOR operation

02 10ASCII Data Characters Chechsum 03

Example: card number: 62E3086CED

- Output data:36H、32H、45H、33H、30H、38H、36H、43H、45H、44H
- CHECKSUM: (62h) XOR (E3h) XOR (08h) XOR (6Ch) XOR (EDh)=08h

I nogle kilder ses at der kommer 14 byte.

Dvs. at der fra RF-ID-en sendes 12 Byte. Byte nummer 1 er??
De næste 10 byte er kortets nummer. Hvert tal er gemt som ASCII. Dvs. at fx et 6-tal sendes som 36h.
Sidste byte er en tjecksum, som giver læse-processoren mulighed for at tjekke, om den har læst korrekt.

Sendes data til LCD – hvordan skrives de så ?? (Char)

Lav et program, der skriver tag-id på PC-skærmen i Debug-vinduet, eller på et LCD-display.
Skriv tag-id på PC-skærmen i Debug-vinduet. Evt. i et LCD.

RC-Servomotor

Styr en servomotor fra en Arduino.

Benforbindelser:

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

Se info om RC-Servomotorer her:

Skriv et program med delays, og test en servomotor.

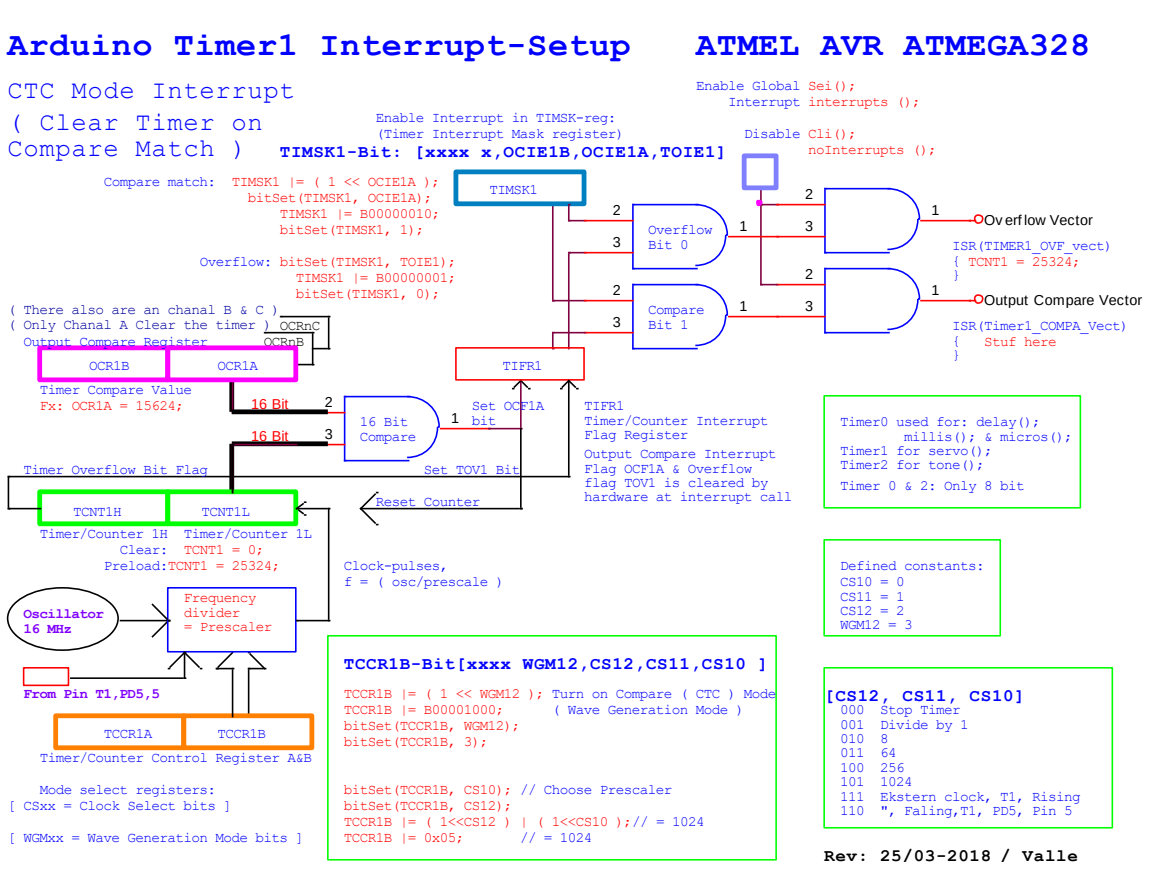
Der skal nok bruges funktionen delayMicrosecond(#);

Men der findes også et bibliotek til Arduino, der kan bruges til at styre servoer direkte.

Brug mit kit og skriv et program til det!! Se mere i Arduino opgaver til elektronik 3. del.

Interrupts:

Følgende diagram er et forsøg på at lave et samlet diagram over en timer og interrupt-struktur



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.



}
```

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;
}
```

Urprogram:

Eksempel:

Krystallets frekvens er 16 MHz. 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 ! – skal også bruges i 3. del opgaverne !

```
/*  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;
  }
}
```

Opgave:

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.

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