

5. November 2019

Udvidet micro:bit kit

Oversigt	3
Målsætninger	3
Specifikationer	3
Grund setup	5
Micro:bit	5
Adafruit DragonTail for micro:bit	6
Breadboard	6
Blink med lysdioder	8
Måling af lys	11
Justering med potentiometer	14
Strømforsyning	17
Hall effect sensor	19
NeoPixels	21
Programoversigt	22

Oversigt

For at give et overblik over kassen indhold, vil der i denne guide blive givet eksempler på opstillinger med hver komponent i kassen, samt en funktionel beskrivelse af hver enkelt.

Målsætninger

1. **At blive fortrolig med kassens indhold:** For at anvende indholdet i en praksis sammenhæng er det i første omgang nødvendigt at blive fortrolig med funktionaliteten af hver enkelt komponent samt hvad det kan anvendes til.
2. **At opnå generel forståelse:** Når funktionaliteten af de enkelte komponenter i kassen er gennemgået, skulle det gerne være muligt for brugeren at sammensætte de forskellige komponenter således at noget nyt skabes.

Specifikationer

For at komme i gang med denne guide vil det være at foretrække hvis brugeren allerede har grundlæggende kendskab til micro:bit samt programmering af denne i blockly.

Herudover vil det være en fordel at have adgang til en lille skruetrækker (ligekærv), samt 2 stk AAA batterier - gerne genopladelige for at skåne miljøet. Igennem denne guide vil der blive anvendt forskellige forsyninger og der vil løbende blive informeret herom.

Til gennemgang af hvert komponent vil der blive anvendt både diagram samt billede af hvordan kredsløbet bygges på et breadboard. Disse vil være lavet i Fritzing. Den kode der skal laves på microbit'en bliver ligeledes vist, både i MakeCode fra Microsoft og JavaScript.

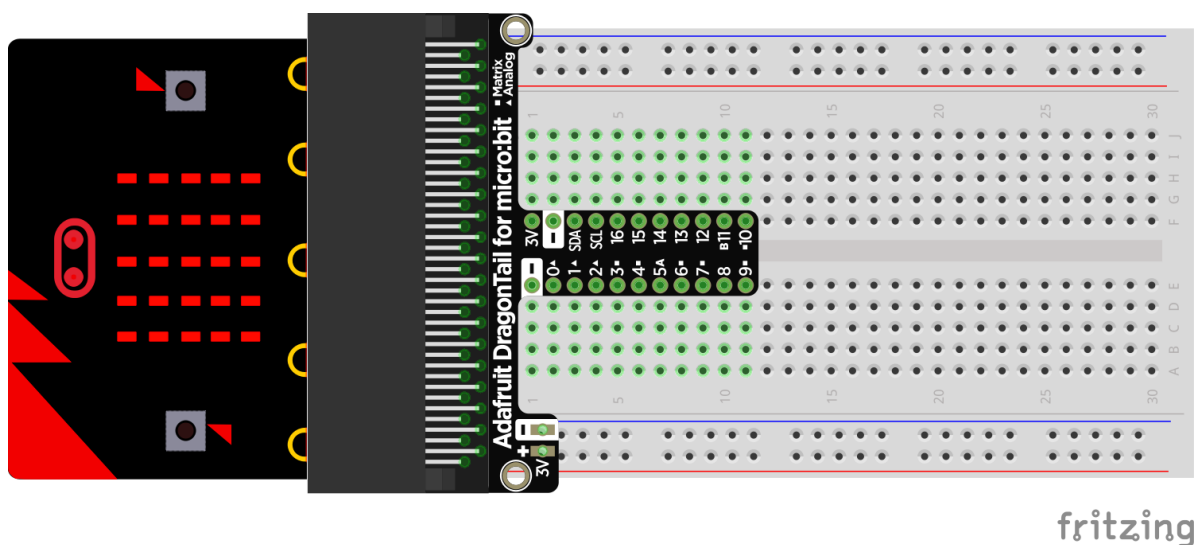
Denne bog er ikke tænkt som en grundbog i hverken elektronik eller programmering, men blot som gennemgang af kassens indhold.

Bagerst i bogen findes en oversigt over links til alle programmer fra MakeCode anvendt i denne bog.



Grund setup

Igennem bogen vil følgende være det grund setup der bliver anvendt. Ved de forskellige eksempler vil der blive beskrevet arbejdsgangen herudfra. Det antages således at brugeren altid tager udgangspunkt herfra.



*Figure 1. Grund setup bestående af en micro:bit, en Adafruit DragonTail
for micro:bit samt et breadboard.*

Grund setuppet består af en micro:bit, en Adafruit DragonTail som forbindender micro:bitten med et breadboard. Vi vil herefter gennemgå enkeltdele i dette grund setup.

Micro:bit

Denne micro:bit er præcis den samme som kendes fra DR Ultra. Den kan som bekendt programmeres enten via MakeCode fra Microsoft eller JavaScript.

Adafruit DragonTail for micro:bit

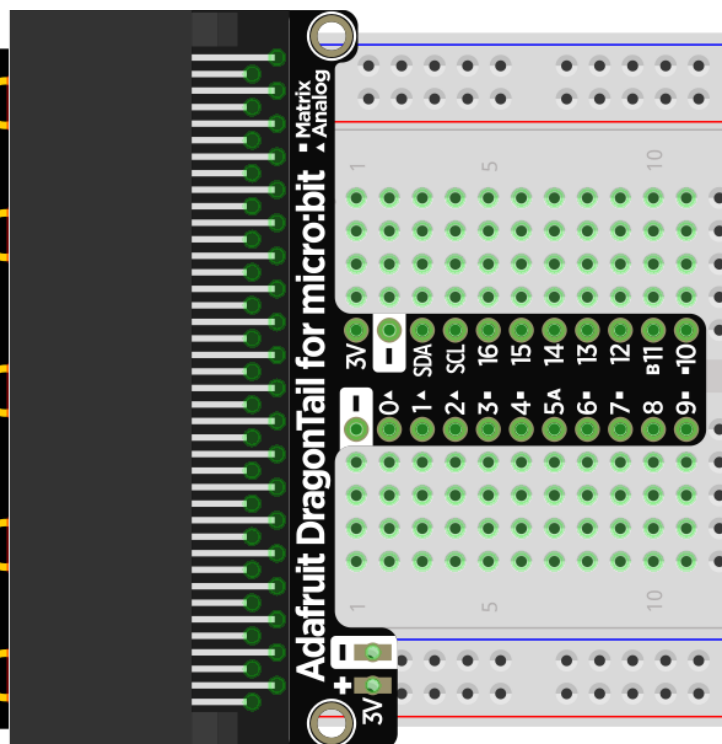
Denne er en connector der forbinder alle forbindelserne i bunden af en micro:bit til rækker på breadboardet. På denne måde bliver det muligt at bruge alle de ekstra I/O (Input/Output) ben der ellers ikke er tilgængelige.

Hvis vi ser nærmere på billedet her til højre, ser vi at flere af benene er markeret med enten en "firkant", en "trekant" eller "A/B".

En markering med en "trekant" betyder at benet kan bruges til analoge signalet.

En markering med en "firkant" indikere at benet kun kan bruge til digitale signaler, ligesom benet også indgår som en del af det matrix display der er på en micro:bit.

Et bogstav "A" eller "B" betyder at benet er forbundet til hhv knap "A" eller "B" på micro:bit'en.

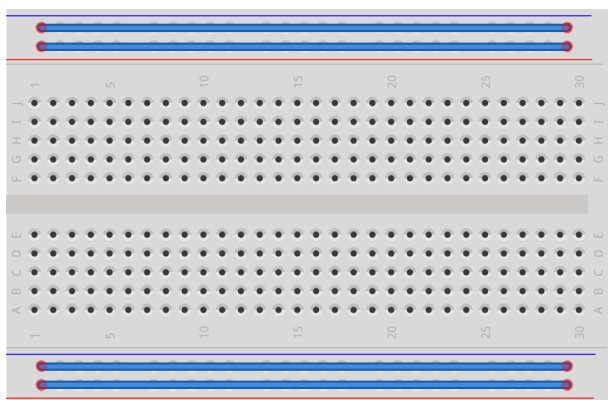


Breadboard

Et breadboard gør det nemt at bygge små kredsløb, uden at man skal lodde. Ydermere kan komponenterne i langt de fleste tilfælde genbruges når de have været en tur i boardet.

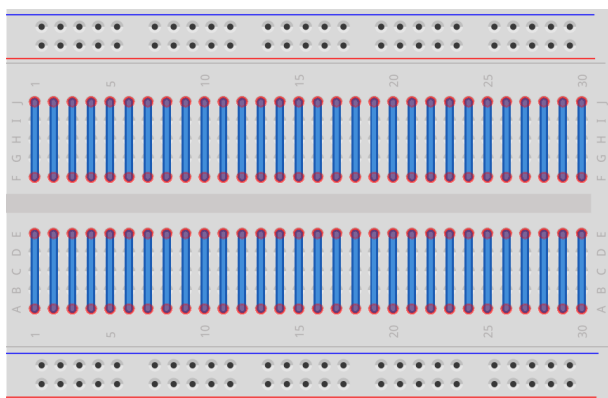
Opbygningen af et breadboard er således at det er opdelt i to hovedområder. Et byggeområde og et forsyningsområde.

Øverst og nederst på breadboardet er to parallelle rækker af huller der løbet i hele boardets længde. Disse fire rækker (2x2) bruges oftest til at have forsyningsspændingen løbende i. Når en Adafruit DragonTain anvendes lægger den 3V fra micro:bit'en på de to nederste rækker på boardet. Husk, at når man sætter en Adafruit DragonTain sammen med et breadboard er det god skik at vende breadboardet således at rækken markeret med "rød" bliver "+" og rækken markeret med "blå" bliver "-". På billedet herunder ses hvordan rækkerne er forbundet i forsyningsområdet.



fritzing

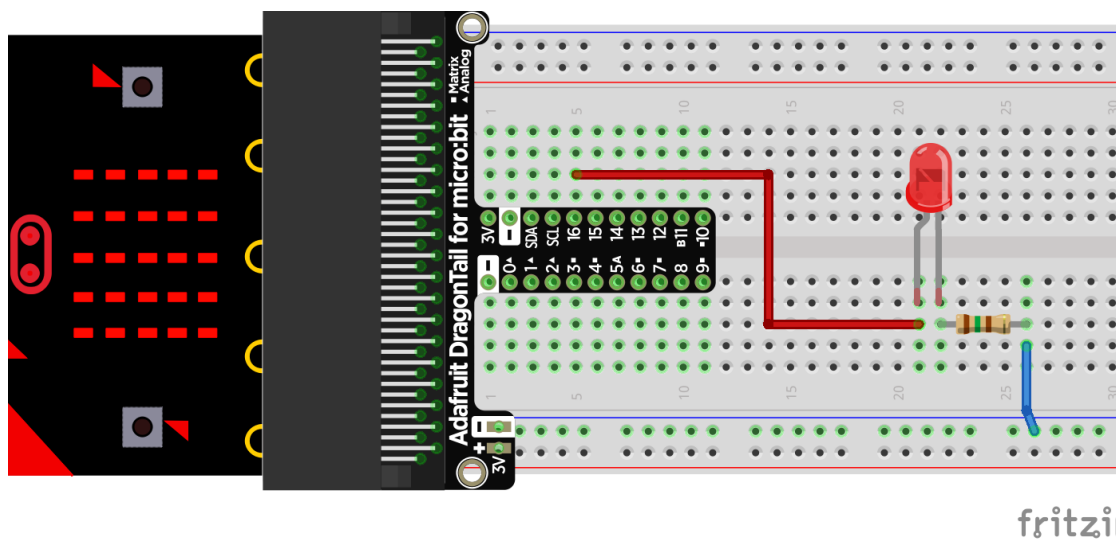
Midterst på breadboardet er byggeområdet. Dette er opdelt i to områder. På tegningen herunder ses hvordan rækkerne er forbundet byggeområdet.



fritzing

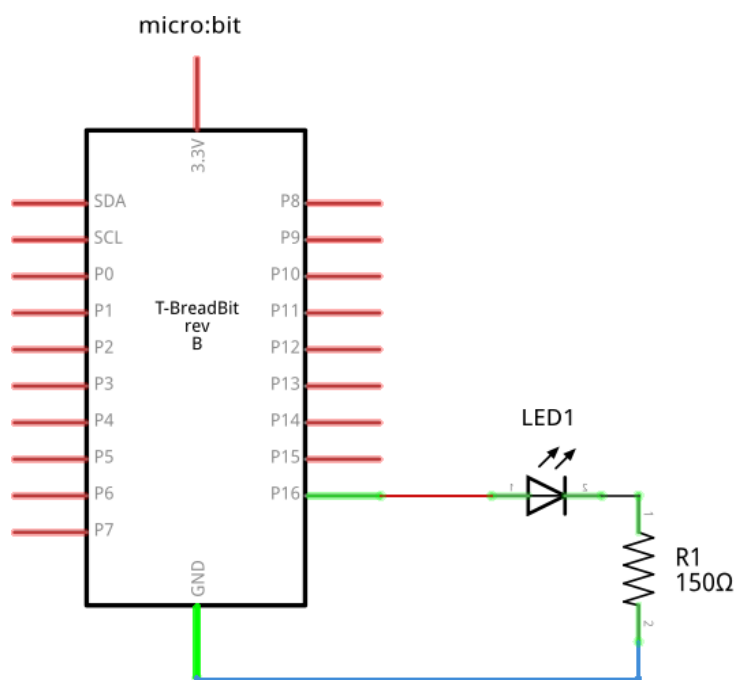
Blink med lysdioder

I kassen findes 5 lysdioder i farverne rød, blå, grøn, gul og hvid. På billedet herunder ses hvordan dioderne skal forbindes til en micro:bit for at vi kan blinke med dem.



fritzi

Diagrammet for ovenstående kredsløb ses herunder.



I kredsløbet er anvendt en 150Ω modstand. Denne modstand anvendes da lysdioderne rød, gul og grøn ikke kan holde til de 3.3V micro:bit'en leverer. Derfor er en strømbegrænsende modstand nødvendig.

For at kunne bestemme modstanden er det nødvendigt at kende lidt data om lysdioderne. I tabellen herunder er de grunddata man normalt vil arbejde ud fra.

Farve	Spænding (spændingsfald)	Strøm
Rød	2,5V	5mA
Gul	2,5V	5mA
Blå	4V	5mA
Grøn	2,5V	5mA
Hvid	4V	5mA

Herefter kan størrelsen på formodstanden for lysdioden bestemmes. Dette gøres ud fra at vi kender den spænding en micro:bit leverer på dens I/O pins samt strøm og spænding (spændingsfald) for lysdioden som angivet i tabellen herover.

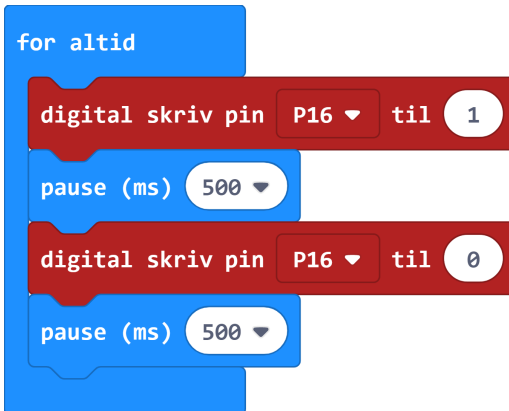
Først omskrives ohm's lov så "R1" isoleres: $U = R1 * I \Leftrightarrow R1 = \frac{U}{I}$

Herefter indsættes således at "R1" bestemmes: $R1 = \frac{3,3V - 2,5V}{5mA} = 160\Omega \approx 150\Omega$

Vi finder her en værdi på 160Ω , hvilket ikke findes i vores kit. Derfor vælges den værdi der ligger tættest på som er i kittet, nemlig 150Ω .

Ved farverne hvid og blå, er denne modstand ikke nødvendig da disse kan holde til en større spænding end hvad en micro:bit leverer. Derfor kan modstanden fjernes og lysdioden forbindes direkte til minus, "-".

Koden der skal til at få lysdioden til at blinke er ganske simpel og vises herunder.



MakeCode - Blockly

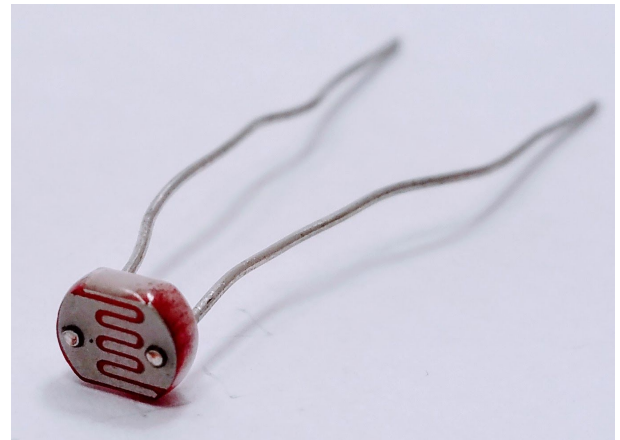
```
basic.forever(function () {  
  pins.digitalWritePin(DigitalPin.P16, 1)  
  basic.pause(500)  
  pins.digitalWritePin(DigitalPin.P16, 0)  
  basic.pause(500)  
})
```

MakeCode - Javascript

Koden herover får lysdioden til at blinke med 1Hz, dvs. at den enten tænder eller slukker hvert halve sekund. Projektet er delt her: https://makecode.microbit.org/_CU4Lh7EJsDMV

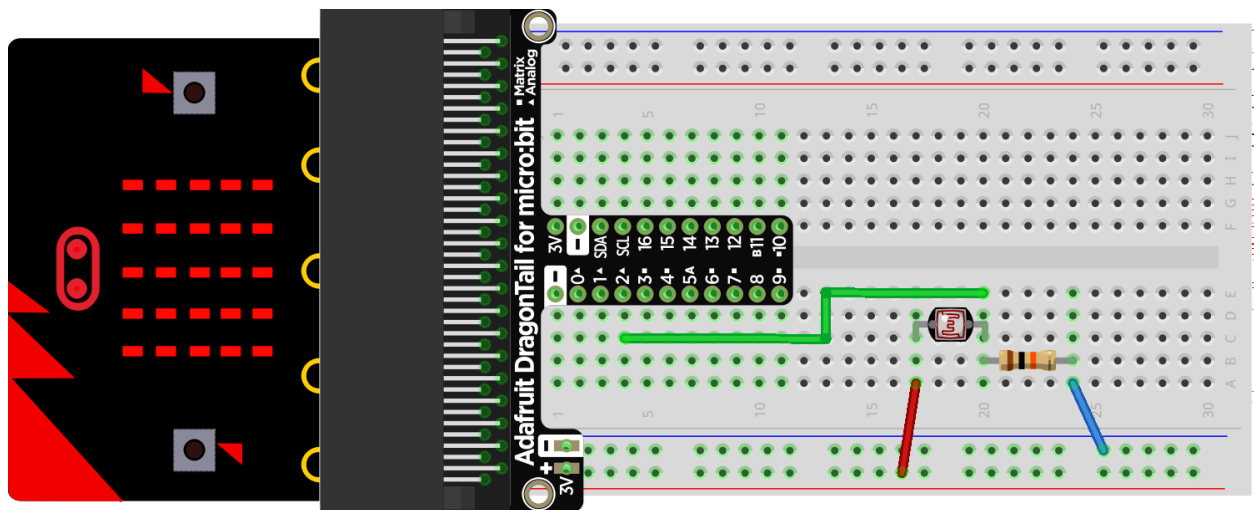
Måling af lys

Vi vil i det kommende afsnit se på hvordan man kan måle på mængden af lys. Til dette formål anvendes en såkaldt LDR modstand. Dette er en type modstand der ændre dens modstand som funktion af lysintensiteten komponentet udsættes for. Det betyder at modstanden falder ved høj lysintensitet og stiger ved lav lysintensitet. Billedet til højre viser en LDR modstand.



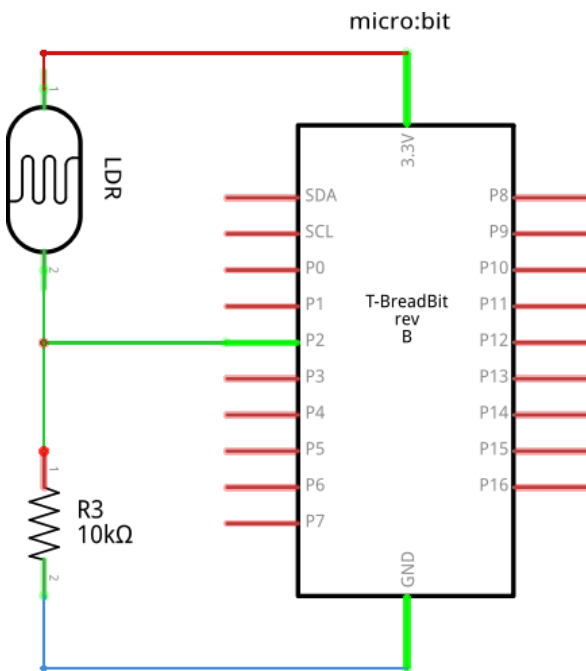
Den LDR i denne kasser er meget lig en PDV-P8001. Søg eventuelt efter datablad på google.

For at kunne anvende en LDR sensor sammen med en micro:bit skal den kobles i et lille kredsløb med en anden fast modstand kaldet en spændingsdeler. Dette gøres for at omsætte modstandsændringen fra LDR modstanden til en spændingsændring som vores micro:bit kan måle. Kredsløbet ses på billedet herunder.



fritzing

Herunder ses diagrammet.



Af diagrammet her til venstre, ses en LDR modstand forbundet i serie med en fast modstand mellem 3.3V og GND. Forbindelsen mellem LDR modstanden og den faste modstand er yderligere forbundet til P2 på vores micro:bit. Denne kobling mellem to modstande kaldes en spændingsdeler og i dette tilfælde er den ene modstand en LDR modstand hvis modstand ændres som funktion af lysintensitet.

For at hente data ind i en micro:bit anvender vi en analog indgang, i dette tilfælde pin 2 på holderen til vores micro:bit som vist på billedet på forrige side samt på diagrammet her ved siden af.



MakeCode - Blockly

```
let lys = 0
basic.forever(function () {
  lys = pins.analogReadPin(AnalogPin.P2)
  basic.showNumber(lys)
})
```

MakeCode - JavaScript

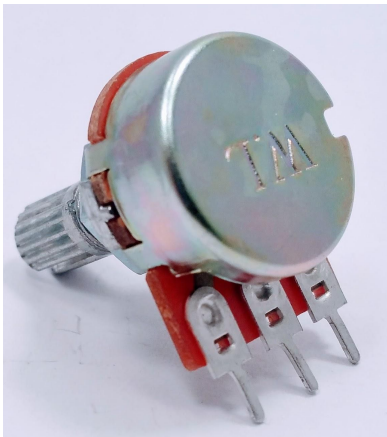
Ovenstående projekt kan findes her: https://makecode.microbit.org/_ipPTC0bqgYm6

Ovenstående program vil udskrive et tal mellem 0 og 1023 som repræsentere mængden af lys der bliver målt. Det er ikke LUX der måles i, men en relativ størrelse, hvis værdi angiver mængden af lys.

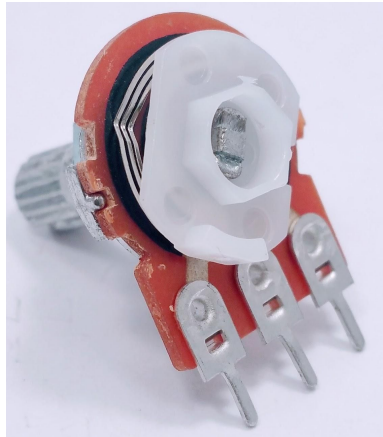
Justering med potentiometer

I sidste opgave arbejdede vi med en variabel modstand kaldet en LDR modstand. I denne opgave skal vi arbejde med en anden type variabel modstand, nemlig et potentiometer.

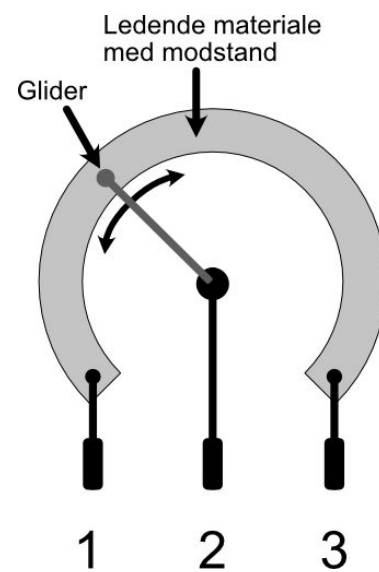
Til forskel fra en LDR modstand der ændre modstand som funktion af lysindfald, så kan vi styre modstanden på et potentiometer. Herunder ses til venstre et almindeligt drejepotentiometer, midterst potentiometer der er åbent og til højre en skematisk tegning af hvordan det virker.



Samlet



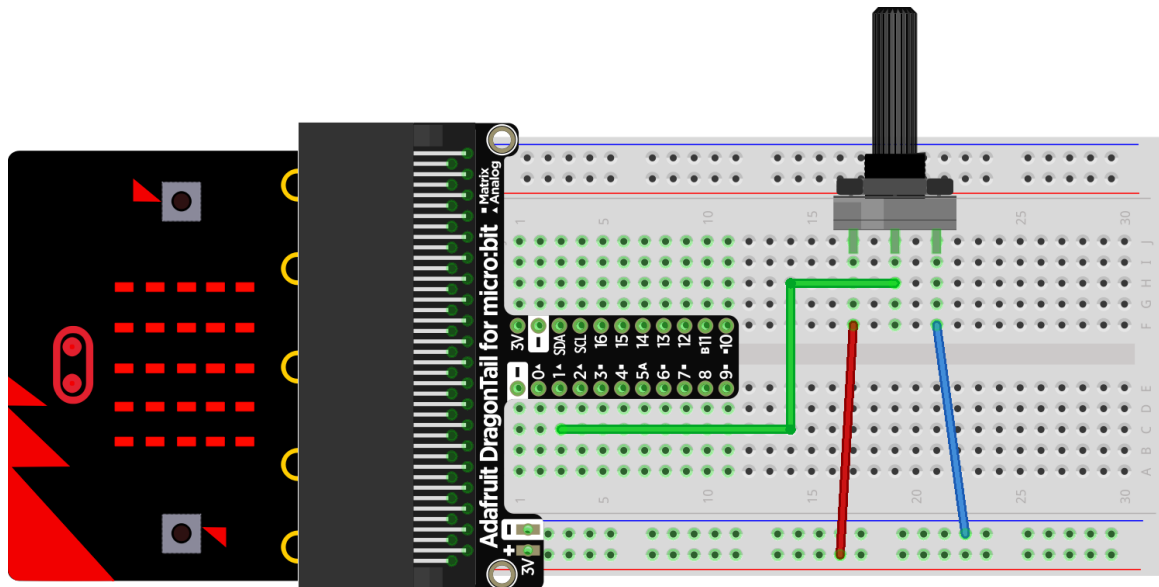
Åbent



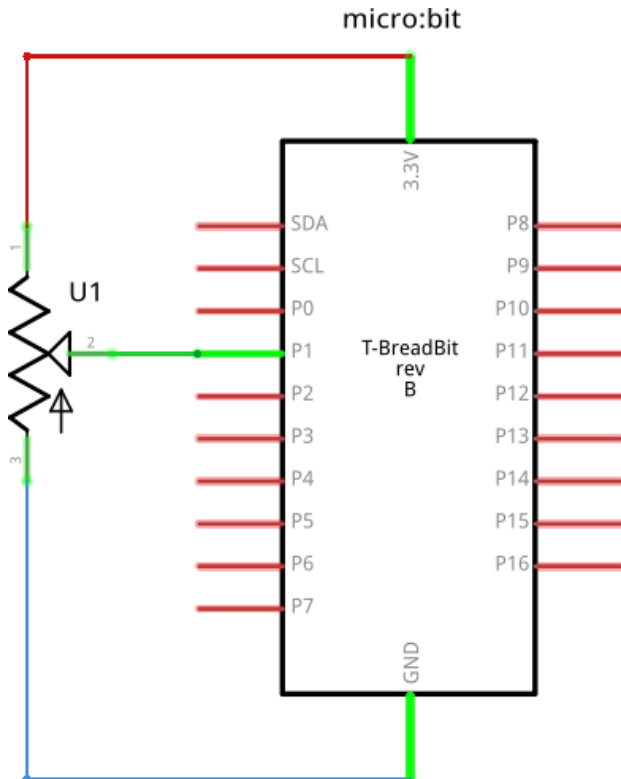
Funktionstegning

Som det kan ses af funktionstegningen herover forbindes ben "1" og "2" af et ledende materiale med modstand. Det betyder at på et potentiometer med en modstand på f.eks. 100K Ω , vil der være en fast modstand mellem ben "1" og "3" på 100K Ω . Den variable del af komponentet måles mellem ben "1" og "2" eller ben "2" og "3" da glideren, hvis position kan justeres ved at dreje på pinden i midten, slutter f.eks. et kredsløb fra ben "1", igennem det ledende materiale til gliderens position og ud gennem ben "2".

I denne lille opstilling ønsker vi at kunne måle ændringen af modstanden på et potentiometer fra en micro:bit. For at kunne dette skal potentiometeret kobles som vist på billede og diagram herunder.



fritzing



På diagrammet herunder ses hvordan potentiometeret er koblet mellem 3,3V fra en micro:bit og GND. Ved at justere glideren - når man drejer på potentiometeret - opnås herefter en spænding på mellem 0V og 3,3V på ben 2, alt efter indstillingen.

Til dette eksempel kan koden fra sidste eksempel anvendes næsten direkte. Dette skyldes at der her også er tale om måling af en varierende analog spænding. Hvis programmet fra sidste eksempel med lys anvendes, vil man få udlæst 0-1023 på displayet på ens micro:bit. Blot skal man ændre

indgangen på ens micro:bit (*Analog pin*) fra "P2" til "P1". Det vil se således ud.



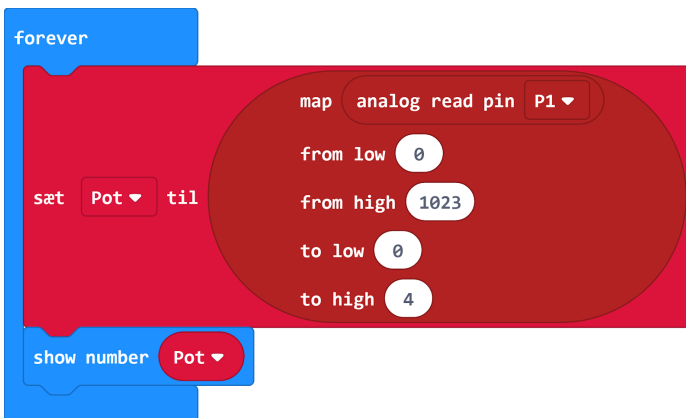
MakeCode - Blockly

```
let Pot = 0
basic.forever(function () {
  Pot = pins.analogReadPin(AnalogPin.P1)
  basic.showNumber(Pot)
})
```

MakeCode - JavaScript

I eksemplet herover er indgangen (*analog pin*) ændret på "P1" og variabelen er omdøbt til "Pot".

Skulle man i stedet ønske at få den reelle spændingen på vist på displayet skal der tilføjes lidt kode. Grunden hertil er at den værdi der ellers vises på displayet er den binære værdi af konverteringen fra analog til digital. Man kan sige det er en værdi er ændre størrelse relativt til ændringen på (*i dette tilfælde*) analog pin "P1". Koden hertil vises nedenfor.



MakeCode - Blockly

```
let Pot = 0
basic.forever(function () {
  Pot = pins.map(
    pins.analogReadPin(AnalogPin.P1),
    0,
    1023,
    0,
    4
  )
  basic.showNumber(Pot)
})
```

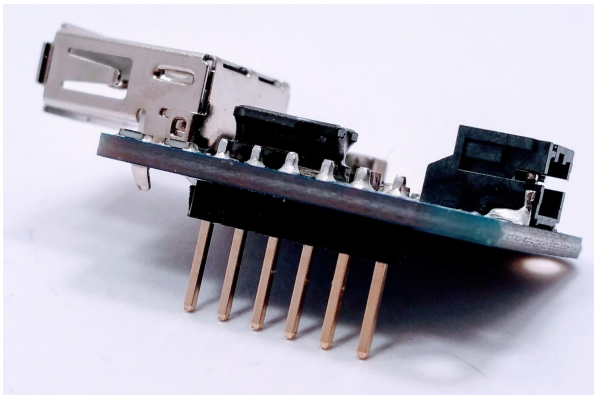
MakeCode - JavaScript

Ovenstående projekt kan hentes her: https://makecode.microbit.org/_5zR3wyapA1f5

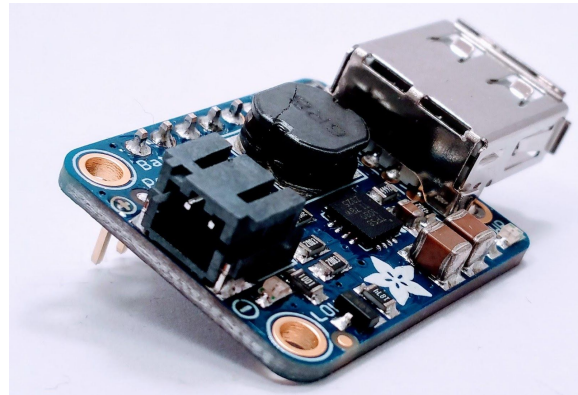
Strømforsyning

I dette afsnit skal vi se på hvad man gør, hvis man har udstyr der kræver mere strøm end hvad en micro:bit kan levere. Faktisk er det langt de fleste ting som falder indenfor kategorien aktuatorer der vil kræve mere strøm. Man kan i grove træk sige at, da aktuatorer påvirker verdenen (får noget til at ske) og det kræver mere energi (strøm) end hvad en micro:bit kan levere.

For at imødekomme denne udfordring er der i kassen inkluderet et separat strømforsyningsmodul.



Strømforsyning - forfra



Strømforsyning - bagfra

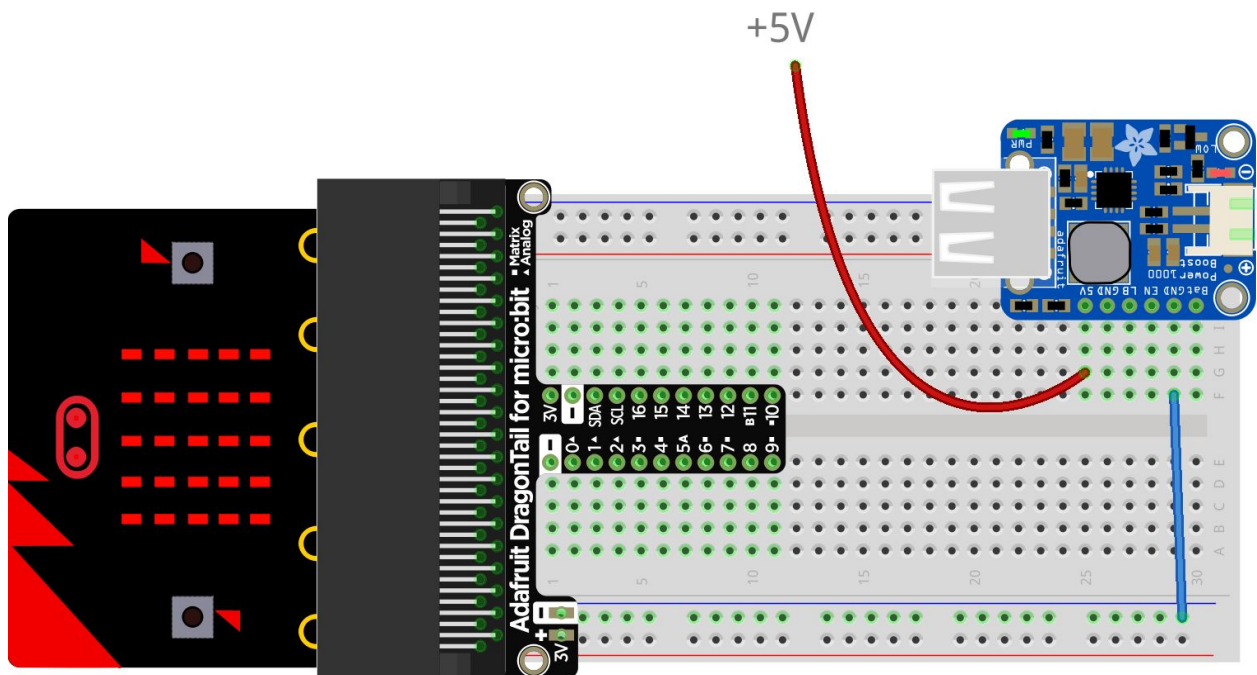
Dette strømforsyningsmodul kan sammen med batteripakken, der også er i kassen, levere en stabil 5V's forsyning på op til 1A. Dette er langt mere end de godt 0,07A (absolut maks) som en micro:bit kan levere.

Når man skal anvende en ekstern strømforsyning, er der et par ting man skal være opmærksom på, for at ens kredsløb kommer til at virke korrekt. Det første er fælles GND. Denne er meget vigtig da man ellers ikke vil have en fælles reference.

Forestil dig et klassisk fysik forsøg hvor hastigheden af et objekt skal bestemmes. Dette gøres også ud fra en fælles reference, nemlig hastigheden "0" hvor både objektet og måleren står

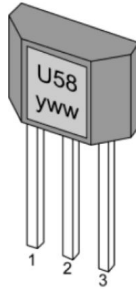
stille. Forestil dig nu at både objekt og måler bevæger sig med forskellig og ukendt hastighed. Så bliver det svært at fastslå hastigheden af objektet.

Af samme årsag, er det første man skal gøre at sikre fælles GND. Herefter er det muligt at anvende den eksterne strømforsyning sammen med en micro:bit uden problemer. På strømforsyningen er også monteret et USB stik, således at strømforsyningen kan forsyne en micro:bit fra batteri, hvis den skal anvendes uden at være koblet til en computer. Herunder er vist hvordan den eksterne strømforsyning kobles.



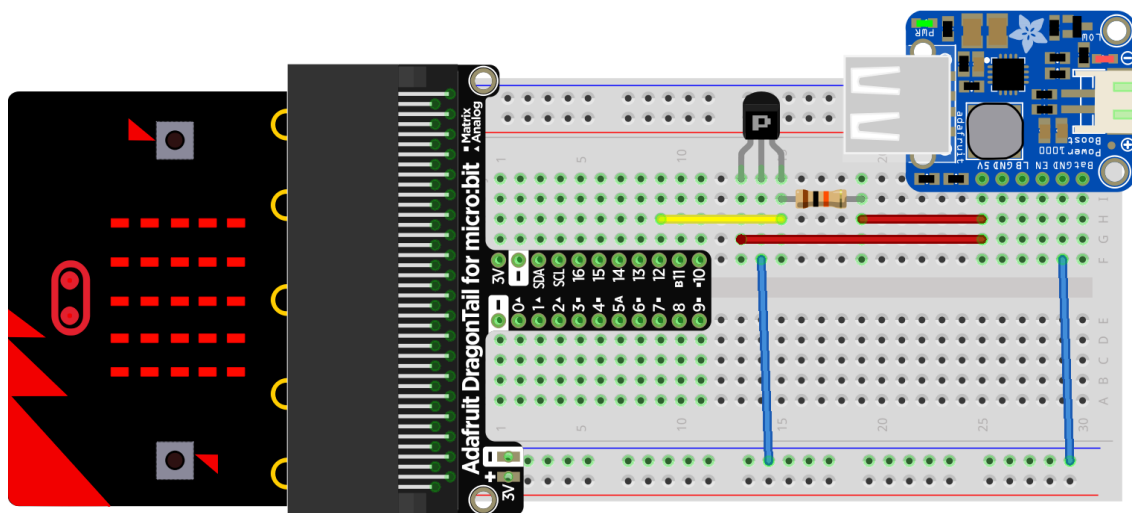
Hall effect sensor

I det følgende vil vi se på, hvordan en hall effect sensor virker. Denne type sensor virker som en kontakt, der kan aktiveres med en magnet. Denne specifikke sensor bliver kun aktiveret ved at bringe "sydpolen" af en magnet nær sensoren. En "nordpol" vil ikke have nogen virkning.



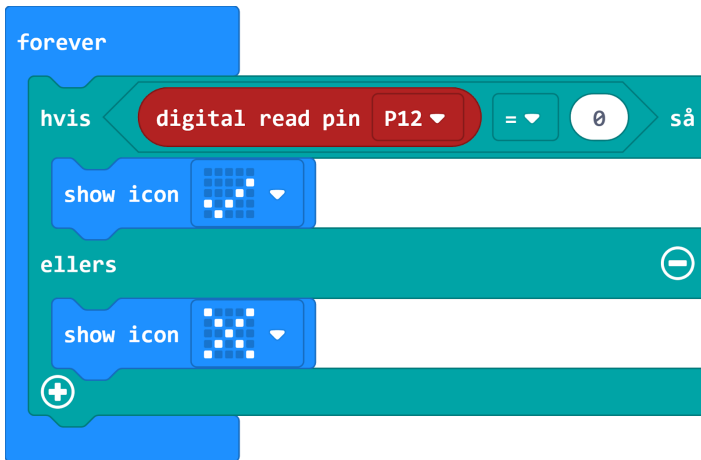
Hall effekt sensor - US5881 - (Billede fra datablad)

For at anvende denne sensor skal opstillingen med strømforsyningen fra sidste afsnit anvendes, da sensoren skal have 5V forsyningsspænding. Tilslutningen skal være som følger: Pin(1) 5V, Pin(2) GND, Pin(3) forbundet gennem en 10KΩ modstand til 5V forsyning samt til Pin(12) på vores micro:bit. Se opstilling herunder.



fritzing

I dette eksempel ønsker vi blot at vise på displayet om sensoren er aktiveret af en magnet eller ej. Koden hertil kan ses herunder.



MakeCode - Blockly

```
basic.forever(function () {  
  if (pins.digitalReadPin(DigitalPin.P12) == 0) {  
    basic.showIcon(IconNames.Yes)  
  } else {  
    basic.showIcon(IconNames.No)  
  }  
})
```

MakeCode - JavaScript

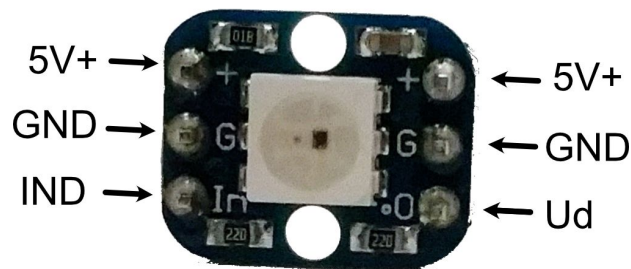
Programmet virker således at et check-mark vises på displayet når en magnets sydpol holdes i nærheden af sensoren. Ellers vises et kryds. Programmet kan hentes her:

https://makecode.microbit.org/_H9fUzHU2gRvs

NeoPixels

For at komme i gang med kode til dette eksempel skal der hentes en udvidelse til MakeCode der hedder "NeoPixel". Herefter vil der være et nyt punkt i block-menuer der hedder "Neopixel".

Men hvad er en Neopixel? Det er et komponent der indeholder 3 lysdioder og en microcontroller. Dette betyder at en neopixel kan vise RGB farver samt at de kan sættes i en række således at man kan styre flere neopixels fra et enkelt I/O ben på ens micro:bit.

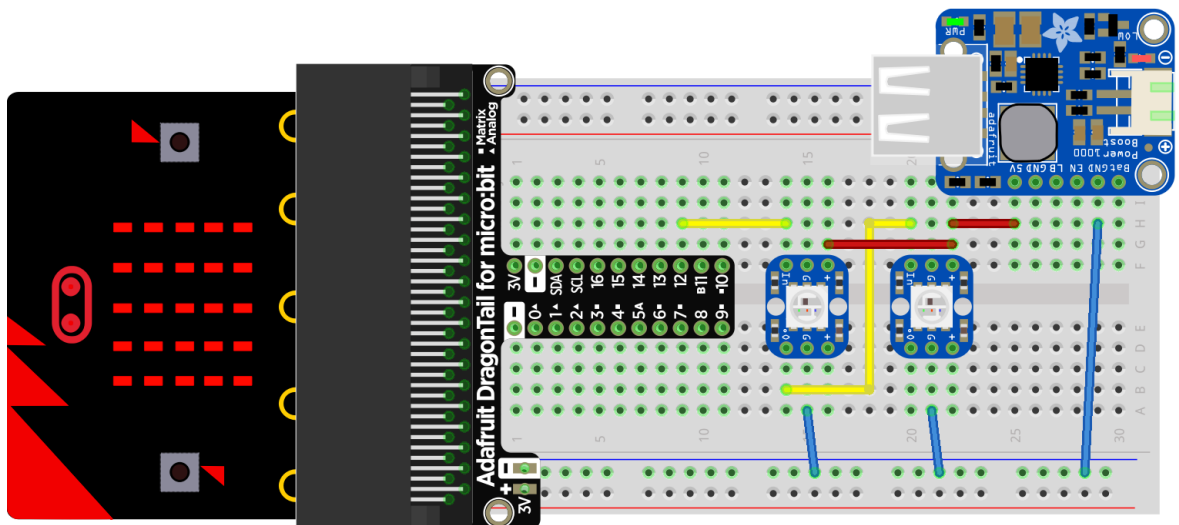


Neopixel

På billedet herover ses en Neopixel. En neopixel har ben på hhv venstre og højre siden. Man skal tænke på det som at signalerne løber fra venstre mod højre - lige som læseretningen.

Forsyningsspænding, hhv. 5V og GND tilkobles i venstre side, og hvis der sidder en neopixel mere i rækken kan forsyningen til den næste hentes fra 5V og GND på højre side. Det nederste ben er kontrol signalet. Som med forsyningsspændingen kommer det ind fra venstre og sendes videre på højre side til den næste neopixel i rækken.

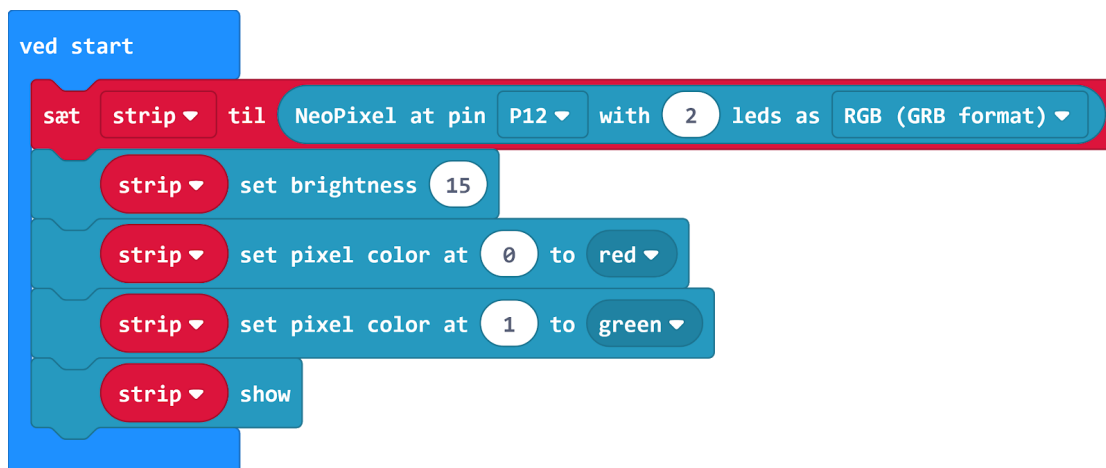
På kredsløbstegningen herunder vises opsætningen af to neopixels i serie. Som tidligere anvendes også her det eksterne strømforsyningsmodul.



fritzing

Byggetegning. Her anvendes "Rød" til 5V, "Blå" til GND og "Gul" til digitalt signal

Når kredsløbet er sat sammen skal koden bygges. I dette eksempel laves blot koden til at tænde de to neopixels med hhv. "Rød" og "Grøn".



MakeCode - Blockly

Vi starter her med at oprette vores række af 2 neopixels i en variabel vi kalder "strip". Herefter sættes "brightness" (lysstyrke). Farven af hhv. neopixel 1 og 2 sættes til "Rød" og "Grøn" og slutteligt aktiveres det hele med "show" kommandoen.

```
let strip = neopixel.create(DigitalPin.P12, 2, NeoPixelMode.RGB)
strip.setBrightness(15)
strip.setPixelColor(0, neopixel.colors(NeoPixelColors.Red))
strip.setPixelColor(1, neopixel.colors(NeoPixelColors.Green))
strip.show()
```

MakeCode - JavaScript

Ønskes anden farve kan nogle pre-indstillede vælges, og hvis anden ønskes kan man også blande sine egne farver. Kode kan hentes her: https://makecode.microbit.org/_V2ubcyap1cYt

Programoversigt

Blink med lysdiode: https://makecode.microbit.org/_CU4Lh7EJsDMV	
Måling af lys: https://makecode.microbit.org/_ipPTC0bqgYm6	
Potentiometer - spænding: https://makecode.microbit.org/_5zR3wyapA1f5	
Hall effekt sensor: https://makecode.microbit.org/_H9fUzHU2gRvs	

NeoPixel:

https://makecode.microbit.org/_V2ubcyap1cYt

