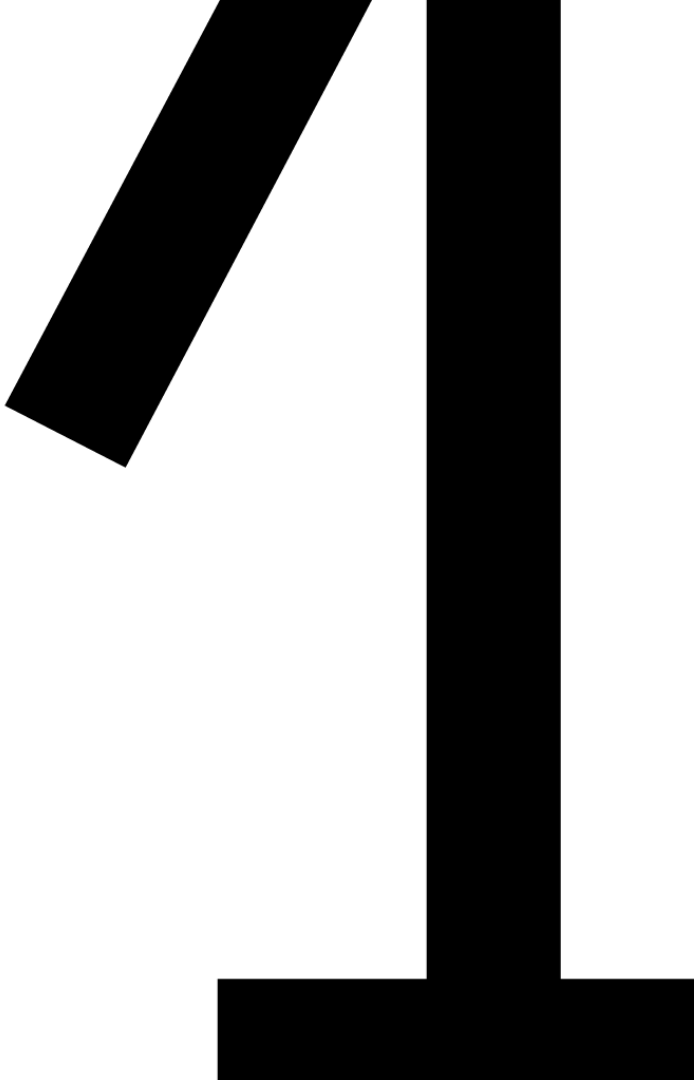
Je eerste mini-satelliet bouwen

Pieter Mestdagh en Julie Willems

KdG Karel de Grote Hogeschool

Belgium esero

Arduinoboard: introductie



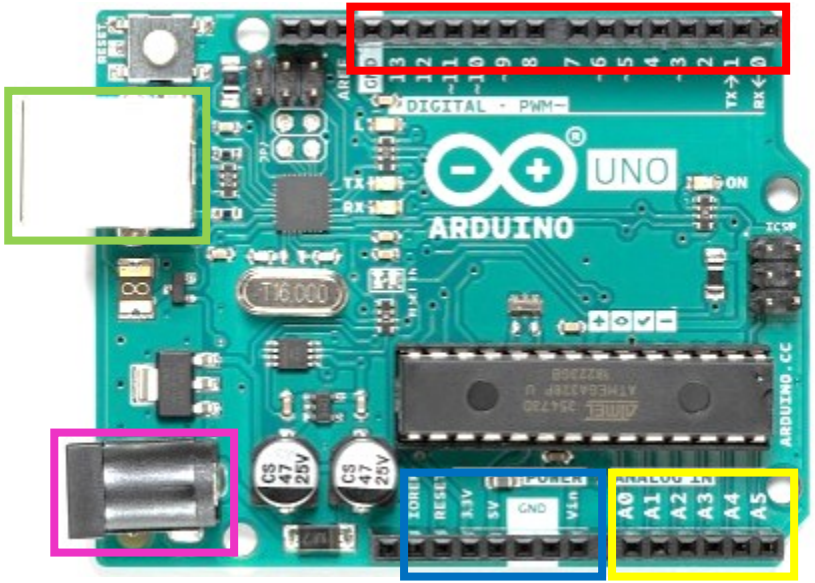
KdG
Karel de Grote Hogeschool

Belgium esero

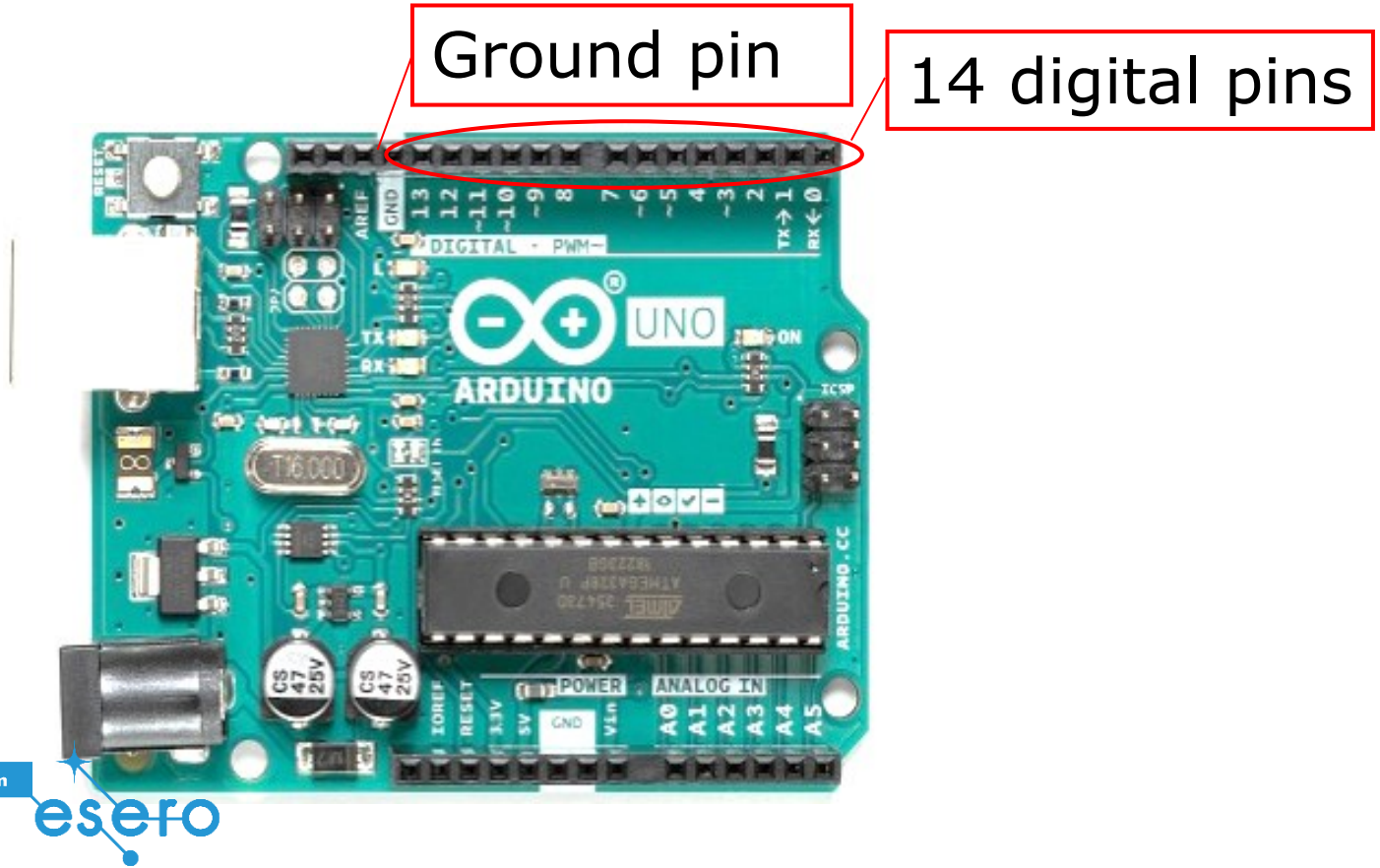
Breadboard Arduino UNO

- Open source

- Rood: 14 Digitale pins voor Input/Output (I/O)
- Geel: 6 Analoge pins voor Input/Output (I/O)
- Groen: USB-verbinding tussen Arduino UNO en pc.
- Paars: Verbinding voor de batterij
- Blauw: Output om externe componenten van stroom te voorzien.



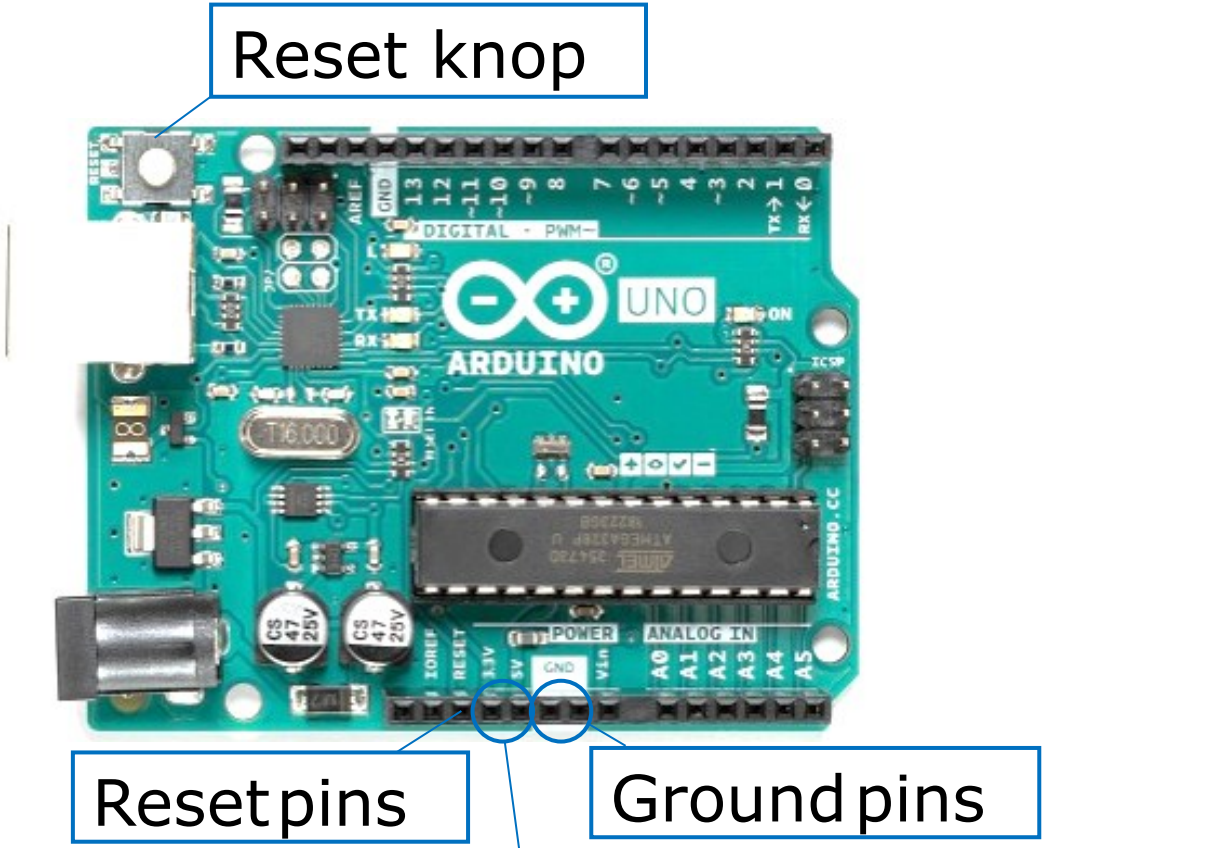
Digitale pins



Analoge pins

6 analoge input pins (5V)

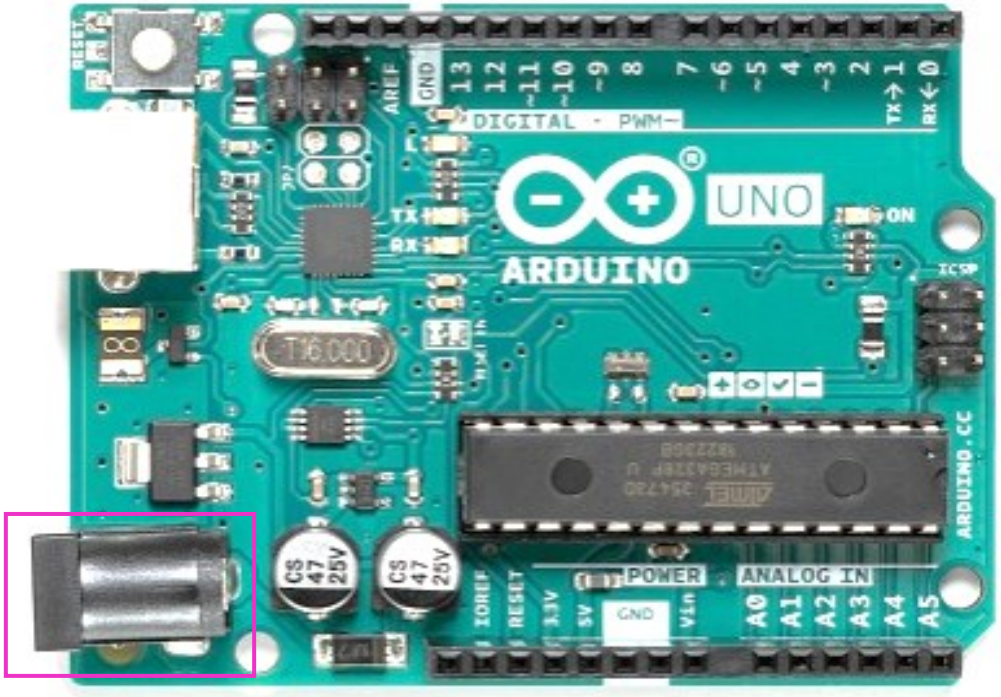
Output


Reset knop
Reset pins
Ground pins
3,3V en 5V output pins
UNO
ARDUINO
DIGITAL · PWM~
POWER
ANALOG IN

DC stroom jack

9V

Power bank (USB)



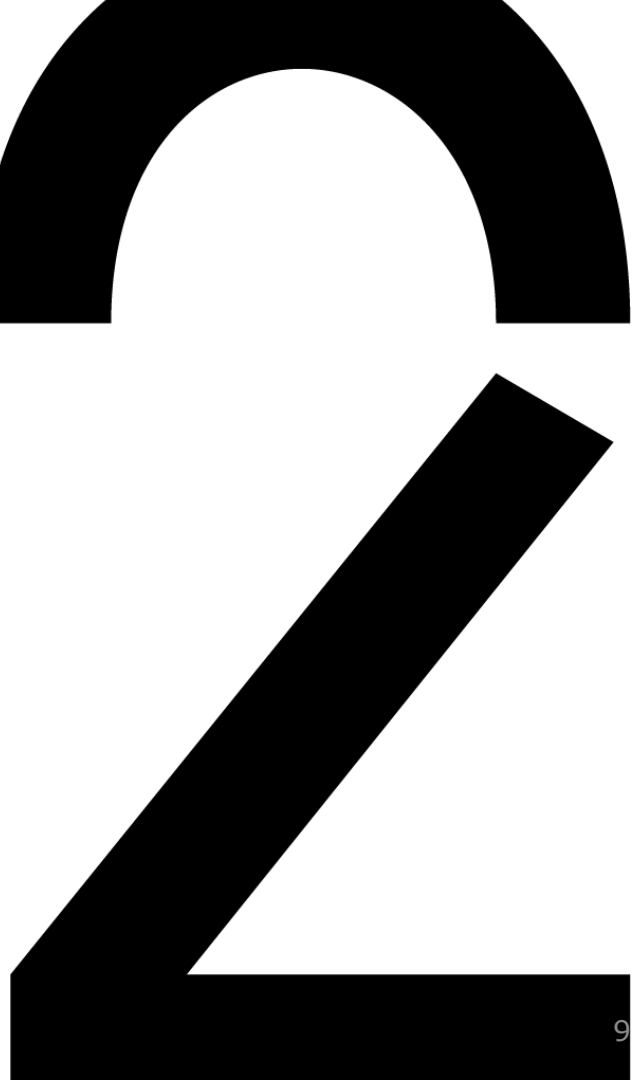
KdG
Karel de Grote Hogeschool

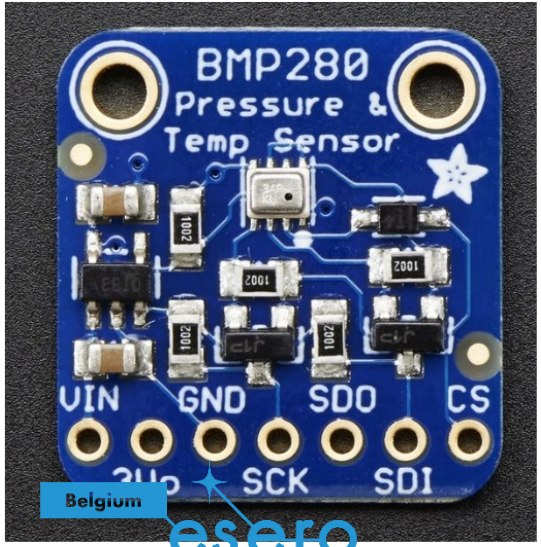
Belgium

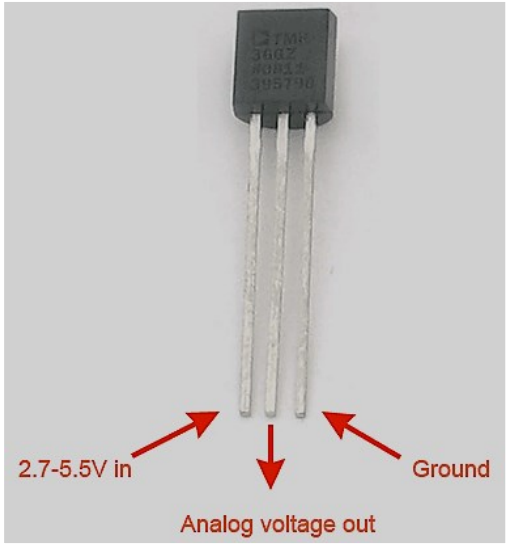
esero

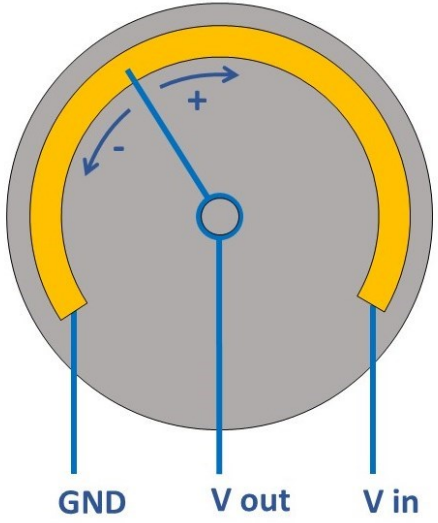
USB-poort

Wat zit er nog in het vormingspakket?







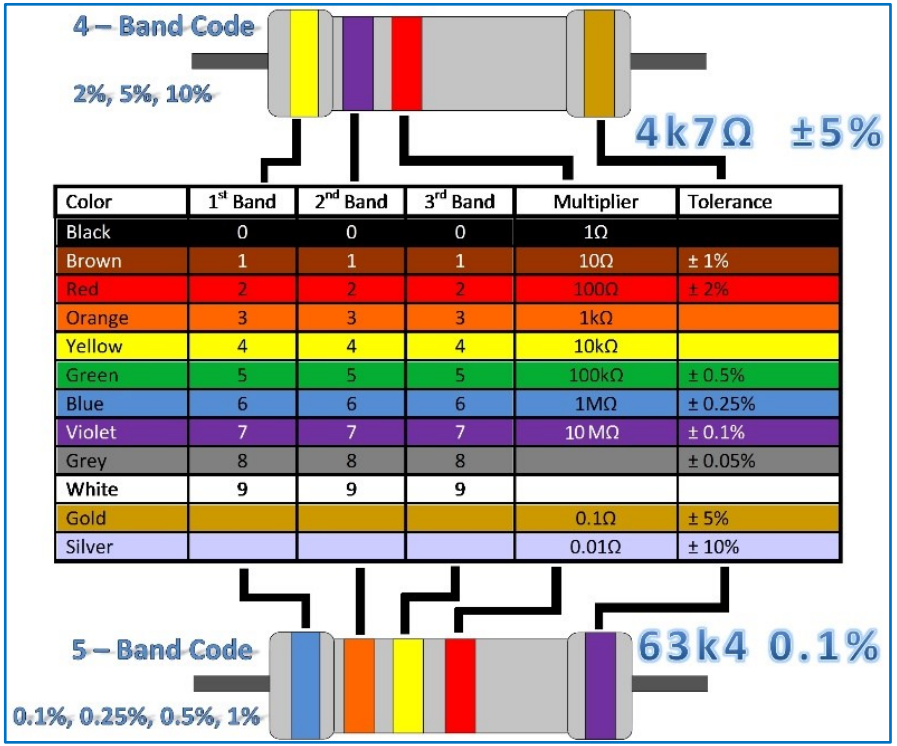


KdG
Karel de Grote Hogeschool

Belgium
esero

Weerstanden

- De eerste twee ringen bepalen het getal.
- De derde ring bepaalt de vermenigvuldigingsfactor, of hoeveel nullen komen achter het getal.
- De vierde ring is om de tolerantie of foutenmarge van de weerstand aan te geven

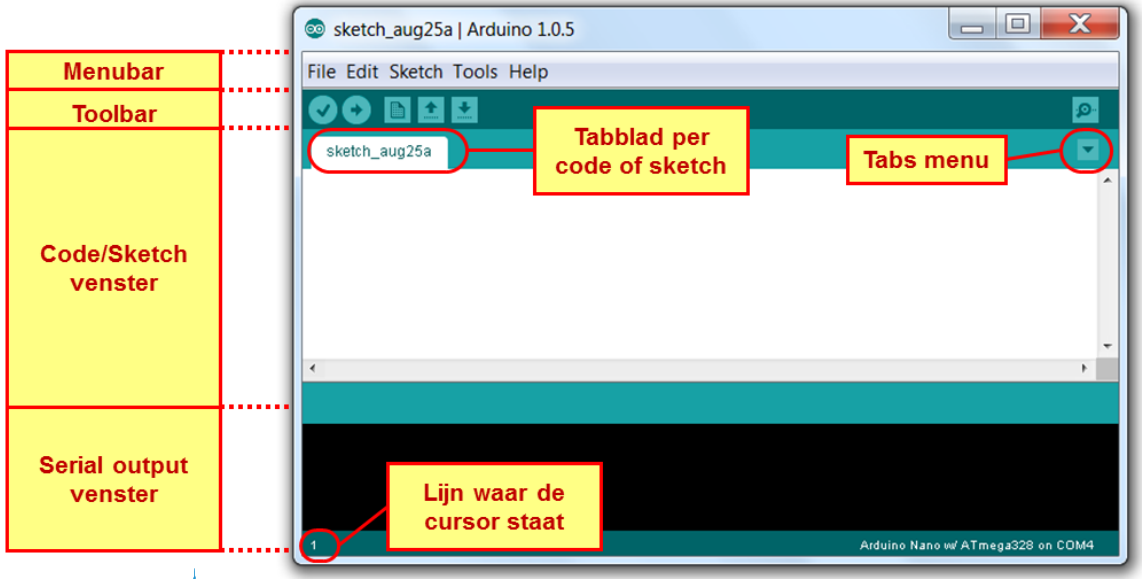


Color	1st Band	2nd Band	3rd Band	Multiplier	Tolerance
Black	0	0	0	1Ω	
Brown	1	1	1	10Ω	± 1%
Red	2	2	2	100Ω	± 2%
Orange	3	3	3	1kΩ	
Yellow	4	4	4	10kΩ	
Green	5	5	5	100kΩ	± 0.5%
Blue	6	6	6	1MΩ	± 0.25%
Violet	7	7	7	10 MΩ	± 0.1%
Grey	8	8	8		± 0.05%
White	9	9	9		
Gold				0.1Ω	± 5%
Silver				0.01Ω	± 10%

Arduino software

Software downloaden + werkwijze

- www.arduino.cc/en/Main/Software



Stappenplan:

1. Verbind het Arduino board
2. Schrijf je code
   Write your code
3. Verify: kijk je code na
4. Upload je code

KdG Karel de Grote Hogeschool

Belgium esero

Programmeertaal

- **Void setup:**

```
Void setup() {Hier schrijf je de code.}
```

Elke code wordt voorafgegaan door de “set up”.

Hier worden bepaalde dingen vastgelegd alvorens aan de operationele code (de code voor de eigenlijke opdracht die je Arduino zal moeten uitvoeren) te beginnen.

Wat er in Set Up staat wordt slechts 1 keer uitgevoerd bij de start van het programma.

- **Void loop:**

```
Void loop() {Hier schrijf je de code.}
```

De loop bevat de eigenlijke opdracht die de Arduino lijn per lijn zal uitvoeren. De loop code wordt continu opnieuw uitgevoerd totdat de gebruiker hem laat stoppen.

- **Commentaar toevoegen**

```
Code // hier schrijf ik uitleg over de code
Nog meer code
```

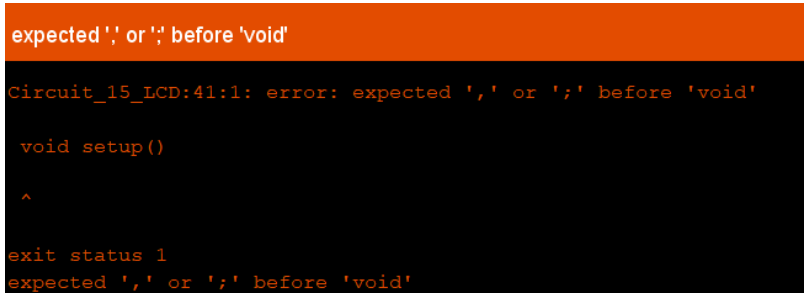
KdG Karel de Grote Hogeschool

Belgium esero

Meest gemaakte fouten

- “;”
- Hoofdlettergevoelig bv. `digitalWrite`

Tip: kopieer en plak je “error” in Google.



Arduino Programming Cheat Sheet

Primary source: Arduino Language Reference
http://arduino.cc/en/Reference/

Structure & Flow

Basic Program Structure

```
void setup() {
  // Runs once when sketch starts
}
void loop() {
  // Runs repeatedly
}
```

Control Structures

```
if (x < 5) { ... } else { ... }
while (x < 5) { ... }
for (int i = 0; i < 10; i++) { ... }
break;     // Exit a loop immediately
continue;  // Go to next iteration
switch (var) {
  case 1:
    ...
    break;
  case 2:
    ...
    break;
  default:
    ...
}
return x;  // x must match return type
return;    // For void return type
```

Function Definitions

```
<ret. type> <name>(<params>) { ... }
e.g. int double(int x) {return x*2;}
```

Operators

General Operators

=	assignment		
+	add	-	subtract
*	multiply	/	divide
%	modulo		
==	equal to	!=	not equal to
<	less than	>	greater than
<=	less than or equal to		
>=	greater than or equal to		
&&	and	||	or
!	not		

Compound Operators

++	increment
--	decrement
+=	compound addition
-=	compound subtraction
*=	compound multiplication
/=	compound division
&=	compound bitwise and
|=	compound bitwise or

Bitwise Operators

&	bitwise and	|	bitwise or
^	bitwise xor	~	bitwise not
<<	shift left	>>	shift right

Pointer Access

& reference: get a pointer
* dereference: follow a pointer

Built-in Functions

Pin Input/Output

```
Digital I/O - pins 0-13 A0-A5
  pinMode(pin,
    [INPUT, OUTPUT, INPUT_PULLUP])
  int digitalread(pin)
  digitalWrite(pin, [HIGH, LOW])

Analog In - pins A0-A5
  int analogRead(pin)
  analogReference(
    [DEFAULT, INTERNAL, EXTERNAL])

PWM Out - pins 3 5 6 9 10 11
  analogWrite(pin, value)

Advanced I/O
tone(pin, freq_Hz)
tone(pin, freq_Hz, duration_ms)
noTone(pin)
shiftOut(dataPin, clockPin,
  [MSBFIRST, LSBFIRST], value)
unsigned long pulseIn(pin,
  [HIGH, LOW])

Time
unsigned long millis()
  // Overflows at 50 days
unsigned long micros()
  // Overflows at 70 minutes
delay(msec)
delayMicroseconds(usec)
```

Math

```
min(x, y)   max(x, y)   abs(x)
sin(rad)    cos(rad)    tan(rad)
sqrt(x)     pow(base, exponent)
constrain(x, minval, maxval)
map(val, fromL, fromH, toL, toH)
```

Random Numbers

```
randomSeed(seed) // long or int
long random(max) // 0 to max-1
long random(min, max)
```

Bits and Bytes

```
lowByte(x)    highByte(x)
bitRead(x, bitn)
bitWrite(x, bitn, bit)
bitSet(x, bitn)
bitClear(x, bitn)
bit(bitn)  // bitn: 0=LSB 7=MSB
```

Type Conversions

```
char(val)     byte(val)
int(val)      word(val)
long(val)     float(val)
```

External Interrupts

```
attachInterrupt(interrupt, func,
 [LOW, CHANGE, RISING, FALLING])
detachInterrupt(interrupt)
interrupts()
noInterrupts()
```

Libraries

Serial - comm. with PC or via RX/TX

```
begin(long speed)  // Up to 115200
end()
int available()  // #bytes available
int read()   // -1 if none available
int peek()   // Read w/o removing
flush()
print(data)     println(data)
write(byte)     write(char * string)
write(byte * data, size)
SerialEvent()  // Called if data rdy
```

SoftwareSerial.h - comm. on any pin

```
SoftwareSerial(rxPin, txPin)
begin(long speed)  // Up to 115200
listen()           // Only 1 can listen
isListening()  // at a time.
read, peek, print, println, write
  // Equivalent to Serial library
```

EEPROM.h - access non-volatile memory

```
byte read(addr)
write(addr, byte)
EEPROM[index]  // Access as array
```

Servo.h - control servo motors

```
attach(pin, [min_uS, max_uS])
write(angle)  // 0 to 180
writeMicroseconds(uS)
   // 1000-2000; 1500 is midpoint
int read()    // 0 to 180
bool attached()
detach()
```

Wire.h - I²C communication

```
begin()      // Join a master
begin(addr) // Join a slave @ addr
requestFrom(address, count)
beginTransmission(addr) // Step 1
send(byte)              // Step 2
send(char * string)
send(byte * data, size)
endTransmission()       // Step 3
int available() // #bytes available
byte receive()  // Get next byte
onReceive(handler)
onRequest(handler)
```

Variables, Arrays, and Data

Data Types

boolean	true | false
char	-128 - 127, 'a' '$' etc.
unsigned char	0 - 255
byte	0 - 255
int	-32768 - 32767
unsigned int	0 - 65535
word	0 - 65535
long	-2147483648 - 2147483647
unsigned long	0 - 4294967295
float	-3.4028e+38 - 3.4028e+38
double	currently same as float
void	i.e., no return value

Strings

```
char str1[8] =
  {'A','r','d','u','i','n','o','\0'};
  // Includes \0 null termination
char str2[8] =
  {'A','r','d','u','i','n','o'};
  // Compiler adds null termination
char str3[] = "Arduino";
char str4[8] = "Arduino";
```

Numeric Constants

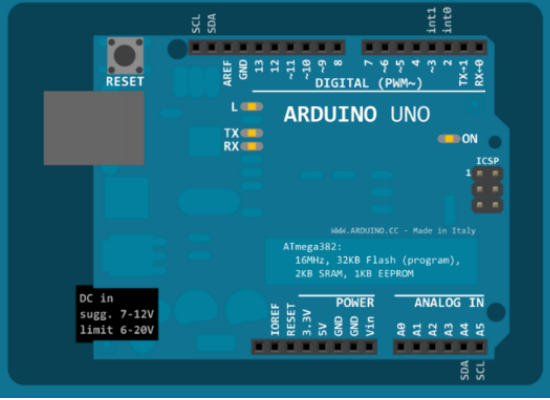
123	decimal
0b01111011	binary
0173	octal - base 8
0x7B	hexadecimal - base 16
123U	force unsigned
123L	force long
123UL	force unsigned long
123.0	force floating point
1.23e6	1.23*10^6 = 1230000

Qualifiers

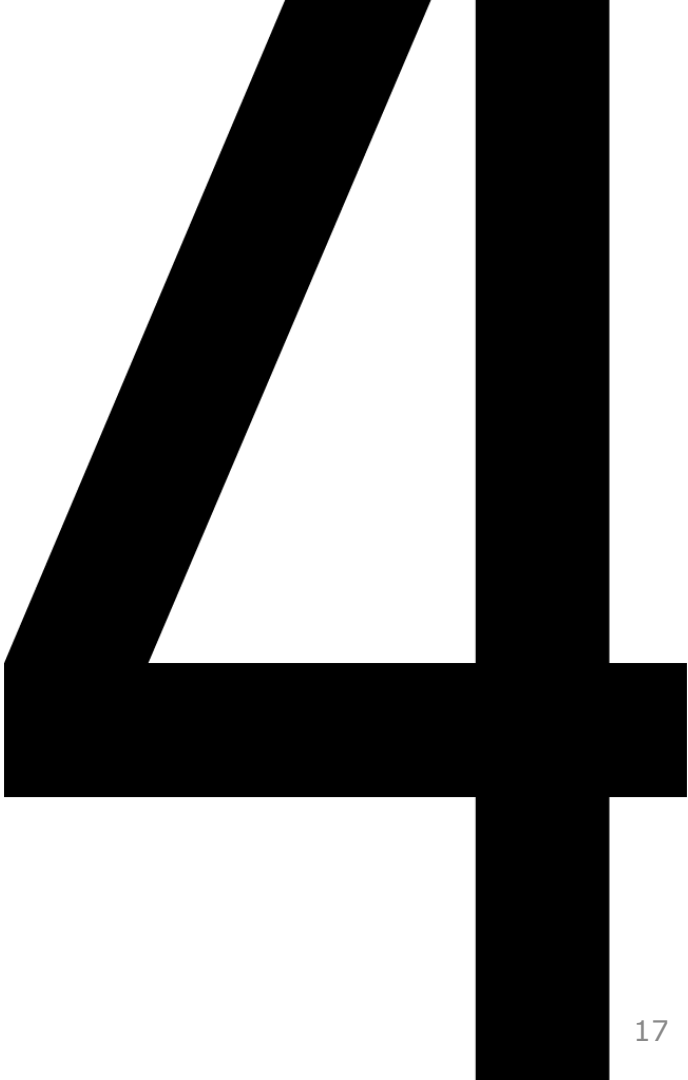
static	persists between calls
volatile	in RAM (nice for ISR)
const	read-only
PROGMEM	in flash

Arrays

```
int myPins[] = {2, 4, 8, 3, 6};
int myInts[6];   // Array of 6 ints
myInts[0] = 42;  // Assigning first
                 // index of myInts
myInts[6] = 12;  // ERROR! Indexes
                 // are 0 though 5
```



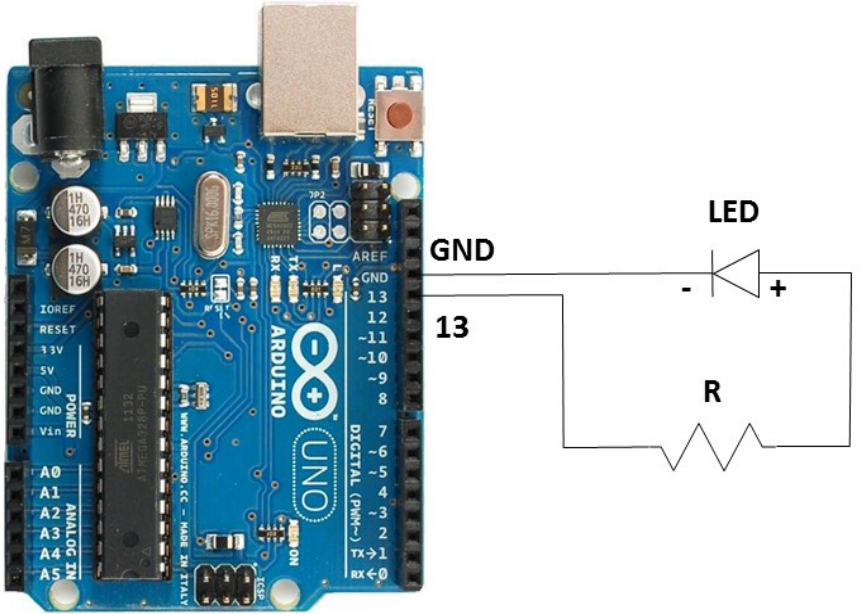
Eenvoudige oefeningen



Een led laten knipperen

Hardware:

- Een Arduino UNO
- Een LED
- Een weerstand van 220 Ω
- Breadboard
- Jumper kabeltjes
- Laptop + USB-kabel



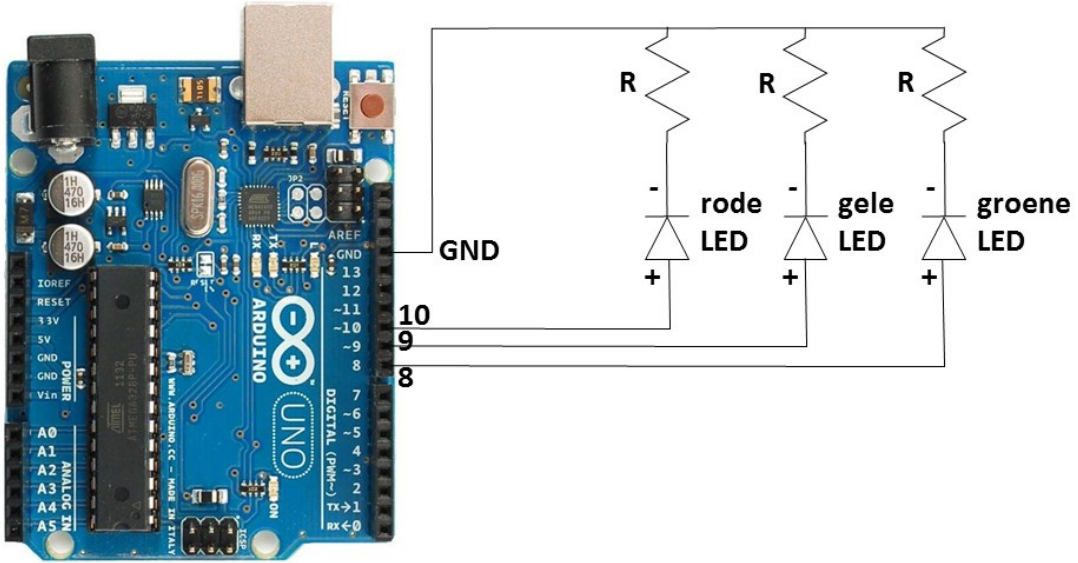
Nood aan een extra uitdaging?

1. *Verander het knipperpatroon.*
2. *Neem een tweede led en sluit aan op pin 12. Laat ze om te beurt knipperen.*

Verkeerslicht

Hardware

- Breadboard
- Rode LED, oranje LED, groene LED
- 3 weerstanden van 220 Ω
- Arduino UNO
- Jumper kabeltjes



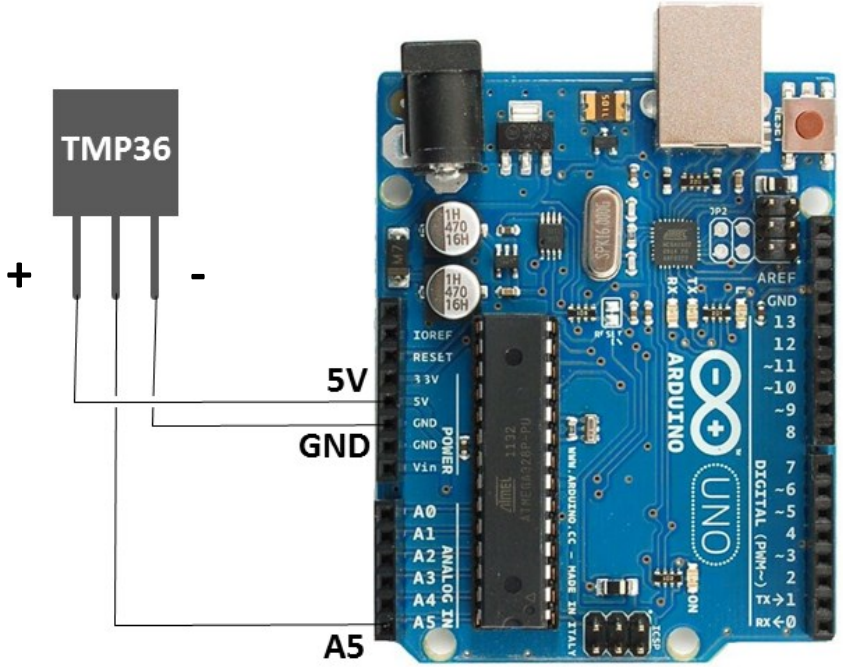
Nood aan extra uitdaging?

1. *Voeg een voetgangersknop toe.*

Temperatuur

Hardware

- 1 Arduino Uno
- 1 breadboard
- 1 temperatuursensor TMP36
- 3 jumper kabeltjes



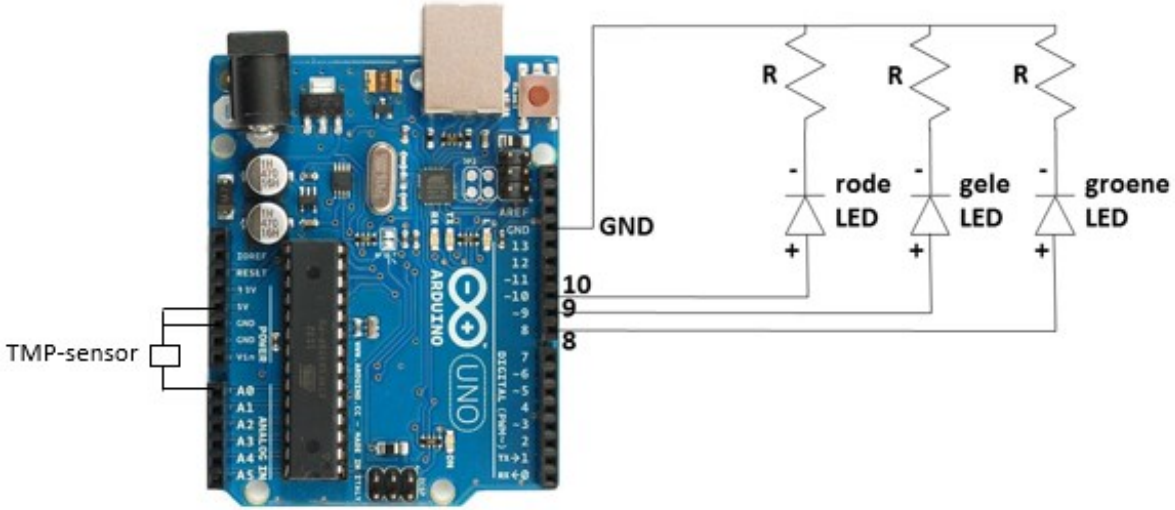
Nood aan extra uitdaging?

1. *Maak een verkeerslicht waarmee je de gewenste temperatuur kan aangeven.*

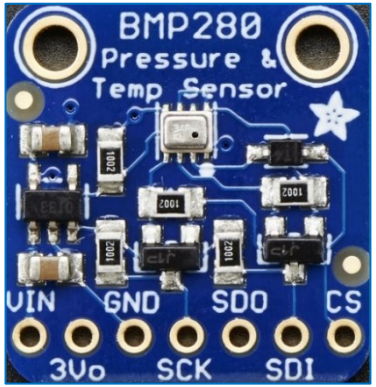
Met verkeerslicht gewenste temperatuur aangeven

Hardware

- Breadboard
- Rode LED, oranje LED, groene LED
- 3 weerstanden (220 Ω)
- TMP-sensor
- Arduino UNO
- Jumper kabeltjes



BMP280 sensor



KdG
Karel de Grote Hogeschool

Belgium
esero

Beschikbare pins

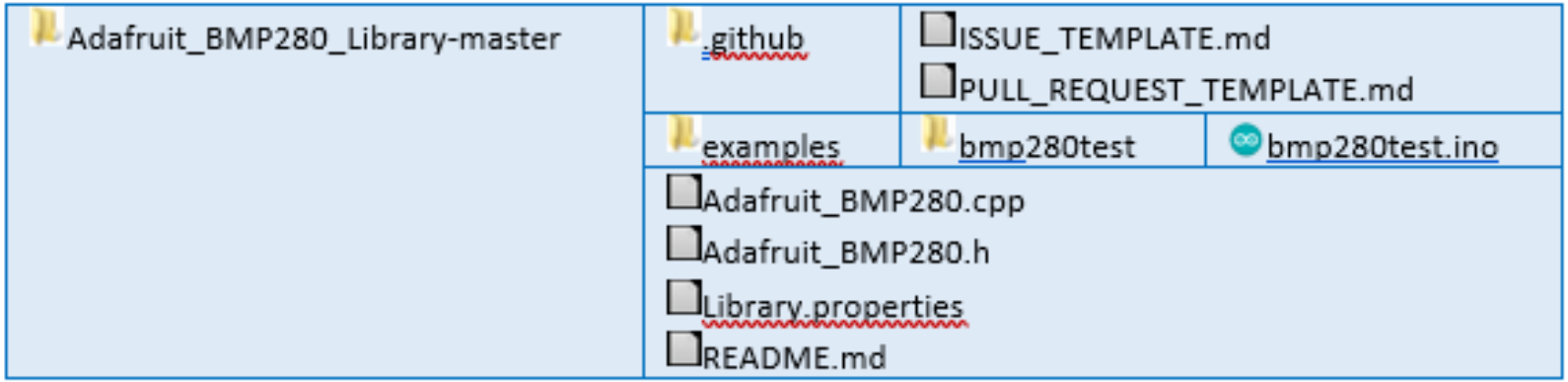
- **Vin:** Hier komt de stroom binnen. De sensor gebruikt 3 Volt DC. Je kan echter even goed 5V op deze pin zetten, want er is een ingebouwde spanningsregelaar.
- **3V0:** Hier kan de stroom buitengaan, met een spanning van 3.3V.
- **GND**

Bij I2C-communicatie:

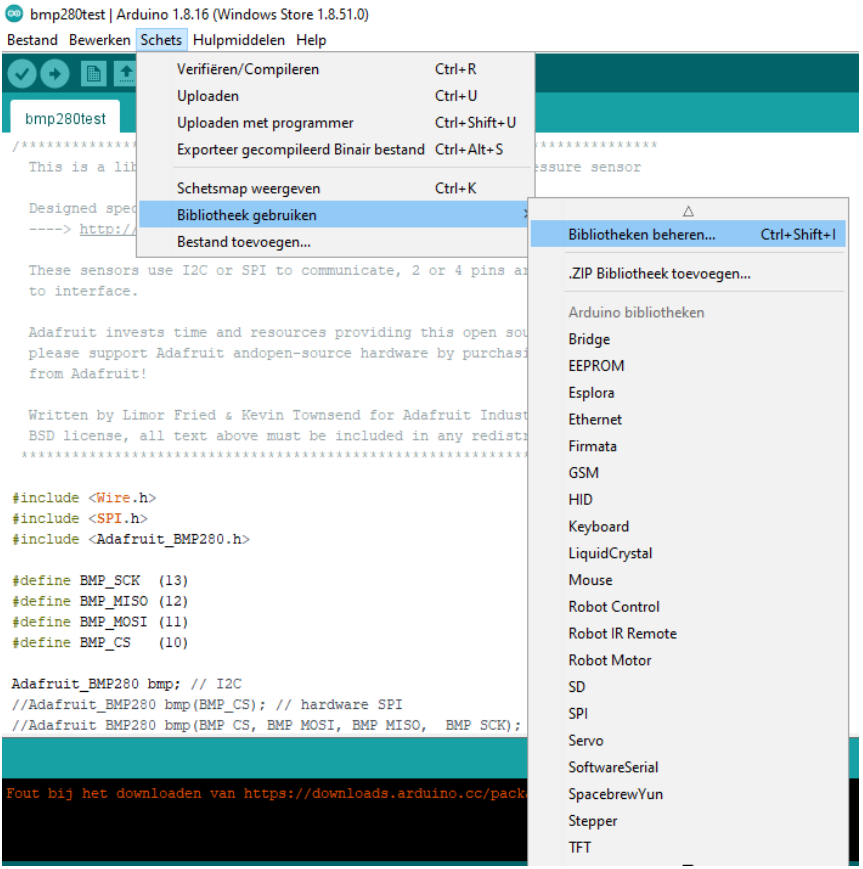
- **SCK:** Serial Clock
- **SDI:** Serial Data In/Out

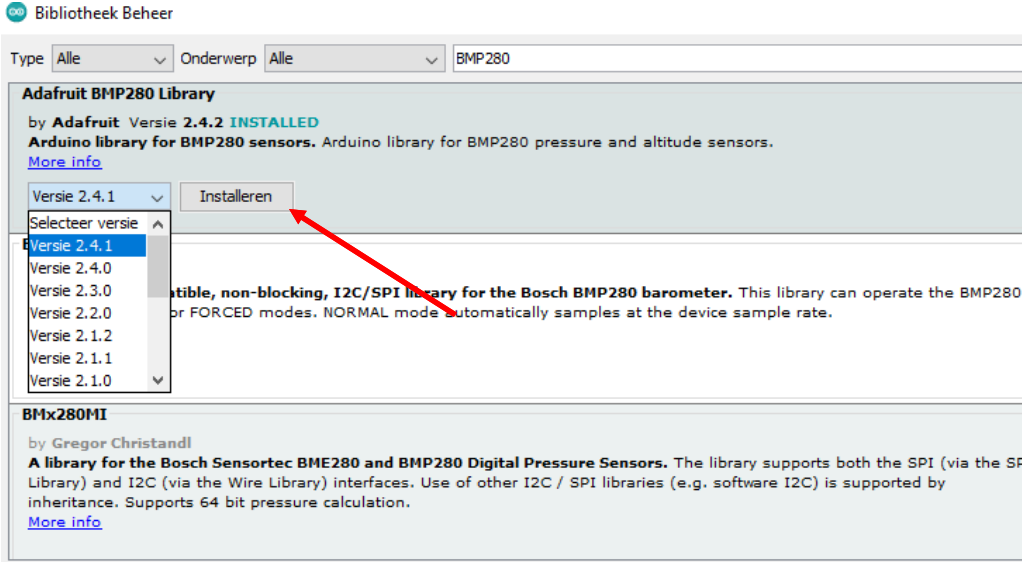
Stappenplan code ophalen

- https://github.com/adafruit/Adafruit_BMP280_Library/archive/master.zip



1. Download de zip-file.
2. Verander zip-bestand van naar, nl. naar Adafruit_BMP280
3. Open je Arduinoprogramma.
4. Schets > Bibliotheek gebruiken > .Zipfile toevoegen
5. Voeg het gedownloade zip-bestand toe.

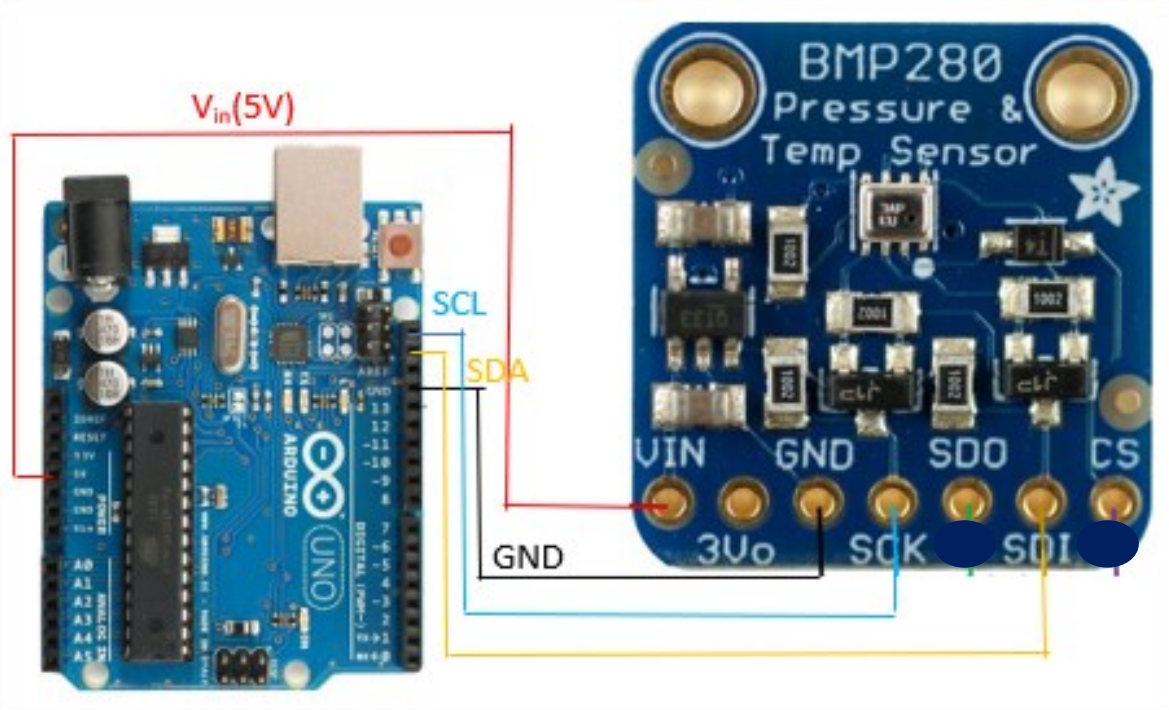

bmp280test | Arduino 1.8.16 (Windows Store 1.8.51.0)
Bestand Bewerken Schets Hulpmiddelen Help
Verifiëren/Compileren Ctrl+R
Uploaden Ctrl+U
Uploaden met programmer Ctrl+Shift+U
Exporteer gecompileerd Binair bestand Ctrl+Alt+S
Schetsmap weergeven Ctrl+K
Bibliotheek gebruiken
Bestand toevoegen...
Bibliotheken beheren... Ctrl+Shift+I
.ZIP Bibliotheek toevoegen...
Arduino bibliotheken
Bridge
EEPROM
Esplora
Ethernet
Firmata
GSM
HID
Keyboard
LiquidCrystal
Mouse
Robot Control
Robot IR Remote
Robot Motor
SD
SPI
Servo
SoftwareSerial
SpacebrewYun
Stepper
TFT
bmp280test
#include <Wire.h>
#include <SPI.h>
#include <Adafruit_BMP280.h>
#define BMP_SCK (13)
#define BMP_MISO (12)
#define BMP_MOSI (11)
#define BMP_CS (10)
Adafruit_BMP280 bmp; // I2C
//Adafruit_BMP280 bmp(BMP_CS); // hardware SPI
//Adafruit_BMP280 bmp(BMP_CS, BMP_MOSI, BMP_MISO, BMP_SCK);
Fout bij het downloaden van https://downloads.arduino.cc/pack


Bibliotheek Beheer
Type Alle Onderwerp Alle BMP280
Adafruit BMP280 Library
by Adafruit Versie 2.4.2 INSTALLED
Arduino library for BMP280 sensors. Arduino library for BMP280 pressure and altitude sensors.
More info
Versie 2.4.1 Installeren
Selecteer versie
Versie 2.4.1
Versie 2.4.0
Versie 2.3.0
Versie 2.2.0
Versie 2.1.2
Versie 2.1.1
Versie 2.1.0
BMx280MI
by Gregor Christandl
A library for the Bosch Sensortec BME280 and BMP280 Digital Pressure Sensors. The library supports both the SPI (via the SP
Library) and I2C (via the Wire Library) interfaces. Use of other I2C / SPI libraries (e.g. software I2C) is supported by
inheritance. Supports 64 bit pressure calculation.
More info

Luchtdruk meten

Hardware

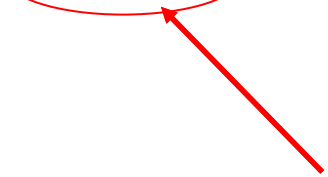
- 1 Arduino Uno
- 1 breadboard + header stacks (6)
- 1 sensor BMP280
- 4 jumper kabeltjes



```
void loop() {
    Serial.print(F("Temperature = "));
    Serial.print(bmp.readTemperature());
    Serial.println(" *C");

    Serial.print(F("Pressure = "));
    Serial.print(bmp.readPressure());
    Serial.println(" Pa");

    Serial.print(F("Approx altitude = "));
    Serial.print(bmp.readAltitude(1013.25)); /* Adjusted to local forecast! */
    Serial.println(" m");
```

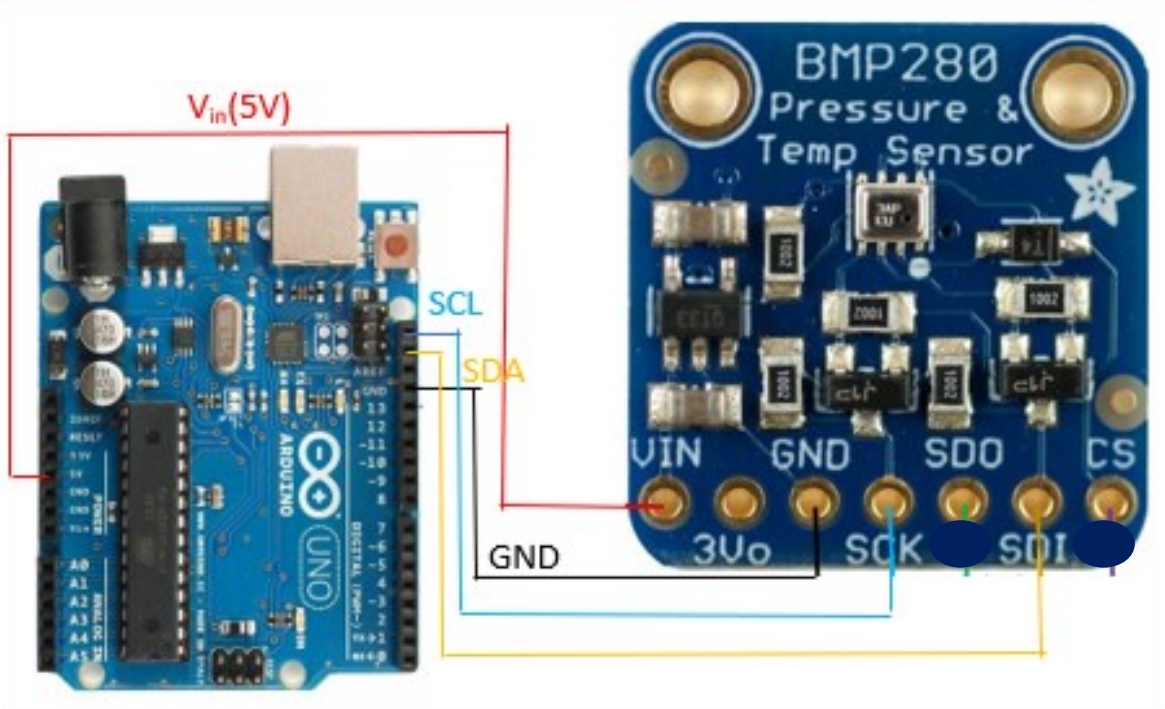


Voeg luchtdruk toe op je huidige positie (opzoeken)

Luchtdruk meten

Hardware

- 1 Arduino Uno
- 1 breadboard + header stacks (6)
- 1 sensor BMP280
- 4 jumper kabeltjes



Nood aan extra uitdaging?

1. *Schrijf de data weg op een SD-kaart.*
2. *Sluit het Arduinoboard aan op een externe voeding (9V).*

Nood aan extra oefening?

- Online code via de Arduino student kit
- Makerspaces.com https://www.makerspaces.com/15-simple-arduino-uno-breadboard-projects/

Solderen

KdG
Karel de Grote
Hogeschool

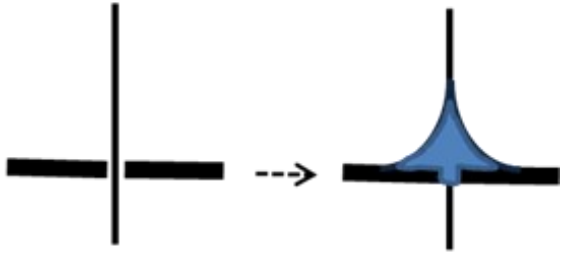
Belgium
esero

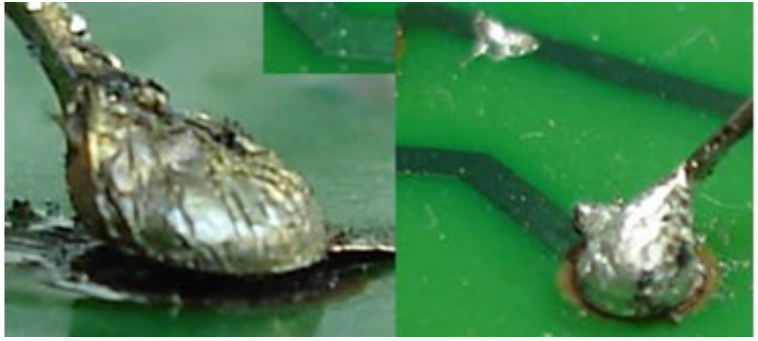
Voorbereiding

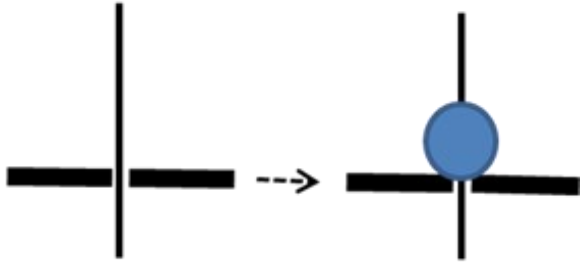
1. Plaats de soldeerbout in zijn standaard en stop de stekker in het stopcontact. De soldeerbout heeft een paar minuten nodig om tot de **juiste temperatuur** van ongeveer 350 - 400°C te komen.
2. Bevochtig de **spons** in de standaard. De spons moet vochtig zijn maar niet kletsnat. Knijp het teveel aan water eruit.
3. Wacht enkele minuten tot de soldeerbout **op werktemperatuur** is. Je kan dit checken door een klein beetje soldeer tegen de punt te houden. Smelt het goed dan is de soldeerbout op de juiste temperatuur.
4. Nu de soldeerbout op temperatuur is veeg je **de punt schoon** op de vochtige spons.
5. **Smelt een klein beetje soldeer** op de soldeertip. Dit wordt vertinnen genoemd en het helpt om de hitte van de punt te verplaatsen naar de te maken verbinding. Dit doe je alleen nadat je de punt hebt schoongemaakt op de spons

Instructies

1. Hou de soldeerbout vast **als een pen**. Doe net alsof je je naam gaat schrijven. Kijk uit dat je het hete gedeelte niet aanraakt.
2. **Raak de plek** waar je de verbinding wil gaan maken aan **met de soldeerpunt**. Zorg ervoor dat de punt zowel het onderdeel als de printbaan of het andere onderdeel aanraakt. Hou de soldeerpunt op deze plek voor 3 seconden en dan....
3. **Plaats wat soldeer** op het te solderen punt. Als alles goed op tempratuur is zal het soldeer soepel vloeien over de printbaan en het component of componenten onderling. Het moet de vorm van een vulkaan aannemen. Voeg soldeer toe tegen de te maken verbinding en niet tegen de soldeerbout.
4. **Haal de soldeer weg** en daarna de soldeerbout. Laat de gemaakte verbinding even afkoelen en houdt de verbinding stil!
5. **Inspecteer** de verbinding goed! Het soldeer moet glimmen en de vorm hebben van een vulkaan. Als dit niet zo is dan moet je de verbinding opnieuw verwarmen en wat meer soldeer toevoegen. Zorg er dit keer voor dat zowel de printbaan als het onderdeel goed heet zijn.







KdG
Karel de Grote Hogeschool

Belgium esero