

Dag1

# Welkom!

Bij de arduino workshop voor beginners



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**  
*Als techniek jouw hobby is!*



Dag1

Welkom

Waar ben ik terecht gekomen

Wie zijn wij :-)



Stichting  
**DE JONGE ONDERZOEKERS  
AMERSFOORT**

Als techniek jouw hobby is!



Dag1

Welkom

Wie zijn jullie  
(kennis en ervaring?)



Stichting  
**DE JONGE ONDERZOEKERS  
AMERSFOORT**

Als techniek jouw hobby is!



Dag1

# Welkom

## Huisregels

- Gebruik materieel
- Hapjes/drankjes
- Wifi:
- SSID: Bitlair-5Ghz
- User: guest / Wachtwoord: eurosnoeren
- (Linux: No CA certificate is required)



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!





Dag1

# Workshop Agenda

**Dag 1 – Intro, theorie en kleine digitale projectjes**

Dag 2 – Theorie en Analoge projecten

Dag 3 – Optional interfaces



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!



# Agenda Dag1

- Wat zit er in het pakket
- Arduino - Algemeen
- Theorie
- Basis kennis programmeren
- basis theorie elektronica
- Praktijk



## Wat zit er in het pakket

- Arduino UNO
- Breadboard
- Jumper wires
- temp sensor
- LDR
- led display
- PIR
- RGB led 10 mm
- Schakelaar
- piezo disc speaker
- gele ledjes
- weerstanden 470, 4K7, 10K



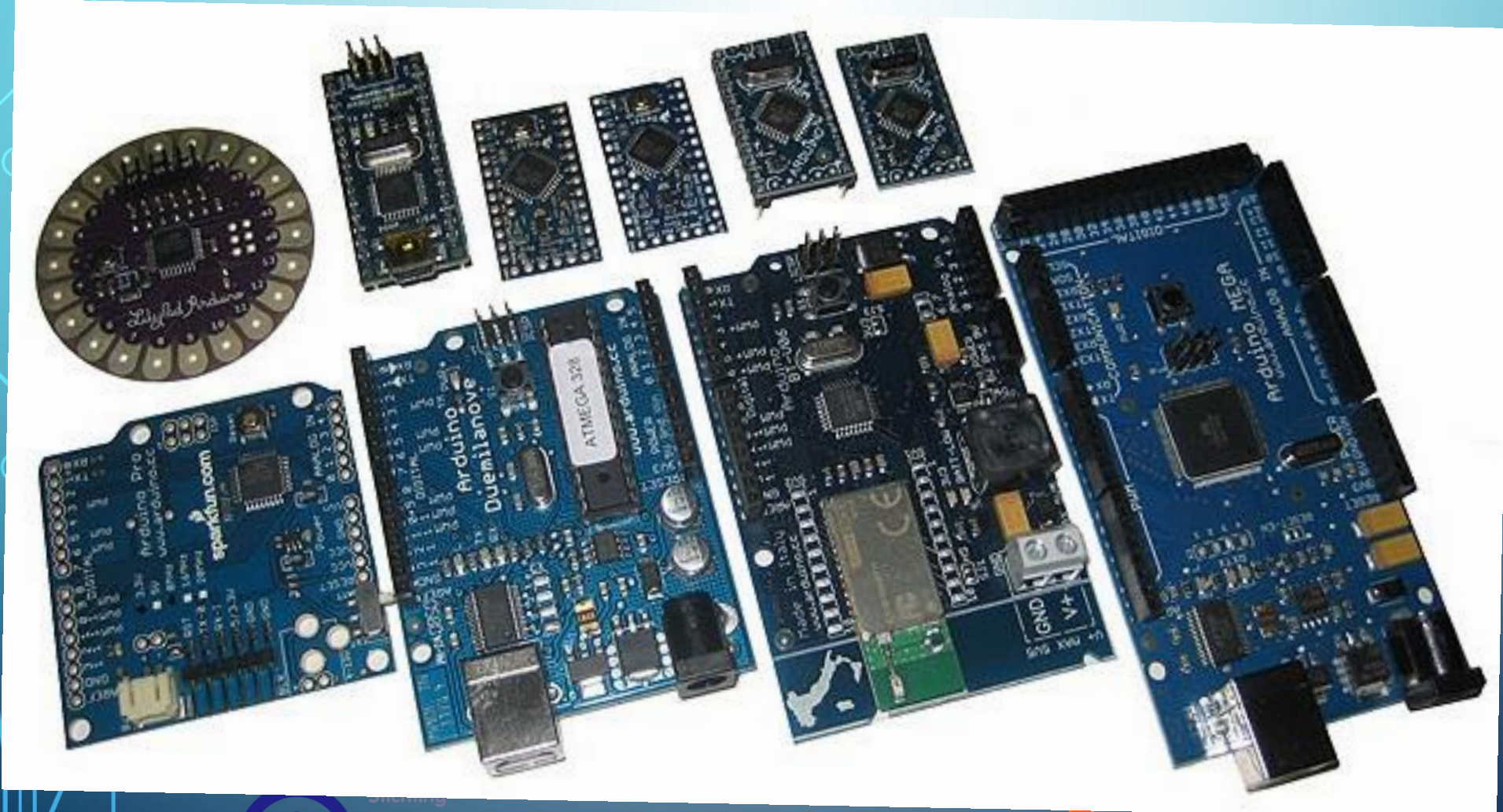
# Arduino

- Soorten
- Eigenschappen
- Cheat Sheet
- info op internet
- Installeren arduino omgeving
- Voorbeelden / libraries





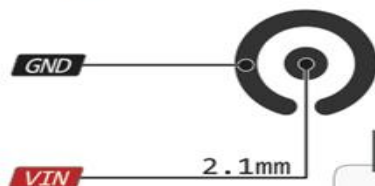
# Arduino (soorten)





# UNO PINOUT

7-12V Depending on current drawn

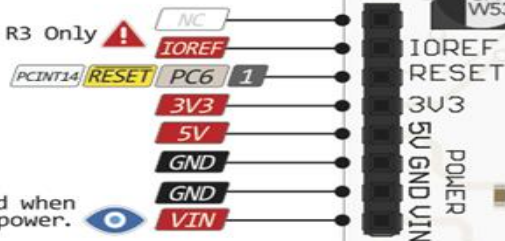


⚠ Absolute MAX per pin 40mA  
recommended 20mA

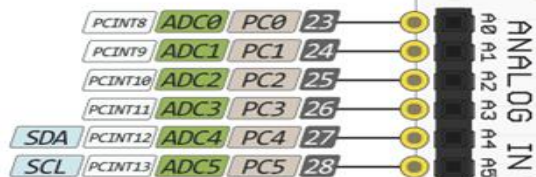
⚠ Absolute MAX 200mA  
for entire package

**IOREF** provides a logic reference voltage for shields that use it. It is connected to the 5V bus.

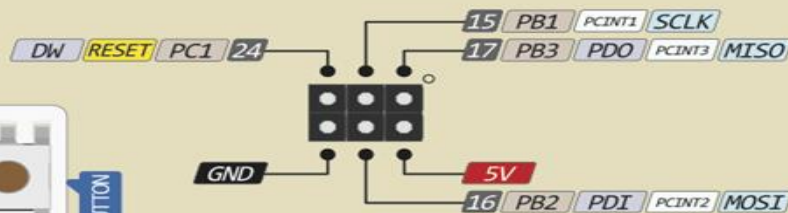
R3 Only ⚠



The input voltage to the board when it is running from external power. Not USB bus power.



ATMEGA 82U/16U2 ICSP



Cut to disable autoreset

DIGITAL

[#=DWM]

TX 1

RX 0

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5

4

3

2

1

0

13

12

11

10

9

8

7

6

5



# ARDUINO CHEAT SHEET V.02B

Mostly taken from the extended reference:  
<http://arduino.cc/en/Reference/Extended>  
 Gavin Smith – Robots and Dinosaurs, The Sydney Hackspace



## Structure

`void setup()` `void loop()`

## Control Structures

```
if (x<5){ } else { }
switch (myvar) {
  case 1:
    break;
  case 2:
    break;
  default:
}
for (int i=0; i <= 255; i++){ }
while (x<5){ }
do { } while (x<5);
continue; //Go to next in do/for/while loop
return x; // Or 'return;' for voids.
goto // considered harmful :-)
```

## Further Syntax

```
// (single line comment)
/* (multi-line comment) */
#define DOZEN 12 //Not baker's!
#include <avr/pgmspace.h>
```

## General Operators

```
= (assignment operator)
+ (addition) - (subtraction)
* (multiplication) / (division)
% (modulo)
== (equal to) != (not equal to)
< (less than) > (greater than)
<= (less than or equal to)
>= (greater than or equal to)
&& (and) || (or) ! (not)
```

## Pointer Access

```
& reference operator
* dereference operator
```

## Bitwise Operators

```
& (bitwise and) | (bitwise or)
^ (bitwise xor) ~ (bitwise not)
<< (bitshift left) >> (bitshift right)
```

## Compound Operators

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

## Constants

```
HIGH | LOW
INPUT | OUTPUT
true | false
143 // Decimal number
0173 // Octal number
B11011111 // Binary (8-bits only)
0x7B // Hex number
7U // Force unsigned
10L // Force long
15UL // Force long unsigned
10.0 // Forces floating point
2.4e5 // 245,000
```

## Data Types

```
boolean (0, 1, false, true)
char (e.g. 'a' -128 to 127)
unsigned char (0 to 255)
byte (0 to 255)
int (-32,768 to 32,767)
unsigned int (0 to 65535)
word (0 to 65535)
long (-2,147,483,648 to 2,147,483,647)
unsigned long (0 to 4,294,967,295)
float (-3.4028235E+38 to 3.4028235E+38)
double (currently same as float)
sizeof(myint) // returns 2 bytes
```

## Strings

```
char S1[15];
char S2[8]={'a','r','d','u','i','n','o'};
char S3[8]={'a','r','d','u','i','n','o','\0'};
//Included \0 null termination
char S4[]="arduino";
char S5[8]="arduino";
char S6[15]="arduino";
```

## Arrays

```
int myInts[6];
int myPins[] = {2, 4, 8, 3, 6};
int mySensVals[6] = {2, 4, -8, 3, 2};
```

## Conversion

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

## Qualifiers

```
static // persists between calls
volatile // use RAM (nice for ISR)
const // make read-only
PROGMEM // use flash
```

## Digital I/O

```
pinMode(pin, [INPUT,OUTPUT])
digitalWrite(pin, value)
int digitalRead(pin)
//Write High to inputs to use pull-up res
```

## Analog I/O

```
analogReference([DEFAULT,INTERNAL,EXTERNAL])
int analogRead(pin) //Call twice if switching pins from high Z source.
analogWrite(pin, value) // PWM
```

## Advanced I/O

```
tone(pin, freqhz)
tone(pin, freqhz, duration_ms)
noTone(pin)
shiftOut(dataPin, clockPin, [MSBFIRST,LSBFIRST], value)
unsigned long pulseIn(pin, [HIGH,LOW])
```

## Time

```
unsigned long millis() // 50 days overflow.
unsigned long micros() // 70 min overflow
delay(ms)
delayMicroseconds(us)
```

## Math

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

## Random Numbers

```
randomSeed(seed) // Long or int
long random(max)
long random(min, max)
```

## Bits and Bytes

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

## External Interrupts

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

## Libraries:

```
Serial.
  begin([300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, 115200])
  end()
  int available()
  int read()
  flush()
  print()
  println()
  write()
```

```
EEPROM (#include <EEPROM.h>)
  byte read(intAddr)
  write(intAddr,myByte)
```

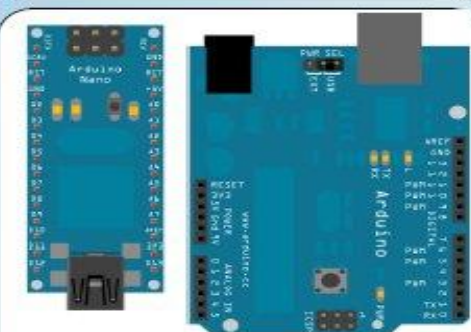
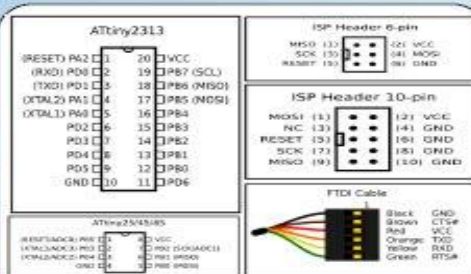
```
Servo (#include <Servo.h>)
  attach(pin, [min_uS, max_uS])
  write(angle) // 0-180
  writeMicroseconds(uS) //1000-2000, 1500 is midpoint
  read() // 0-180
  attached() //Returns boolean
  detach()
```

```
SoftwareSerial(RxPin,TxPin)
// #include<SoftwareSerial.h>
begin(longSpeed) // up to 9600
char read() // blocks till data
print(myData) or println(myData)
```

```
Wire (#include <Wire.h>) // For I2C
begin() // Join as master
begin(addr) // Join as slave @ addr
requestFrom(address, count)
beginTransmission(addr) // Step 1
send(mybyte) // Step 2
send(char * mystring)
send(byte * data, size)
endTransmission() // Step 3
byte available() // Num of bytes
byte receive() //Return next byte
onReceive(handler)
onRequest(handler)
```

	ATmega168	ATmega328	ATmega1280
Flash (2K for bootloader)	16KB	32KB	128KB
SRAM	1KB	2KB	8KB
EEPROM	512B	1KB	4KB

	Duemilanove/ Nano/ Pro/ ProMini	Mega
# of IO	14 + 6 analog (Nano has 14+8)	54 + 16 analog
Serial Pins	0 - RX 1 - TX	0 - RX1 1 - TX1 19 - RX2 18 - TX2 17 - RX3 16 - TX3 15 - RX4 14 - TX4
Ext interrupts	2 - (Int 0) 3 - (Int 1)	2,3,21,20,19,18 (IRQ0-IRQ5)
PWM pins	5,6 - Timer 0 9,10 - Timer 1 3,11 - Timer 2	0-13 53 - SS 51 - MOSI 50 - MISO 52 - SCK 20 - SDA 21 - SCL
SPI	10 - SS 11 - MOSI 12 - MISO 13 - SCK	
I2C	Analogs 4 - SDA Analogs 5 - SCL	



Pics from Fritzing.Org under C.C. license

# Installeren arduino omgeving

- Installeer software

arduino.cc ==> Download

- Bord kiezen

Arduino/Genuino Uno

- Driver <http://sparks.gogo.co.nz/ch340.html>

- Poort kiezen

windows device manager (devmgmt.msc)

linux /dev/ttyUSBx

/dev/ttyACMx

apple/peer

/dev/cu.usbmodem1421 (Arduino/Genuino Uno)



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!





Dag1

# Voorbeelden / libraries

Open Arduino

Gebruik voorbeelden

Open File-Examples

Gebruik libraries

Open Sketch – Include Library

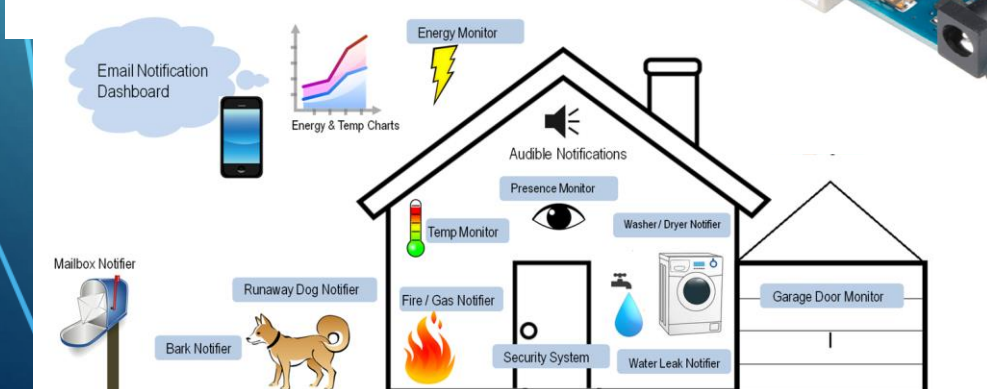
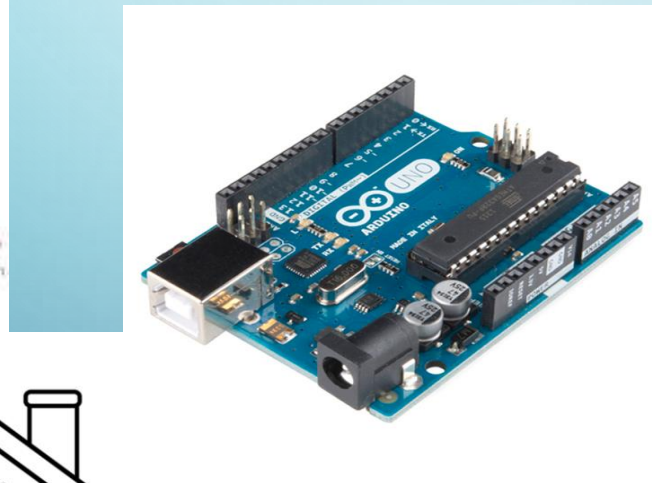
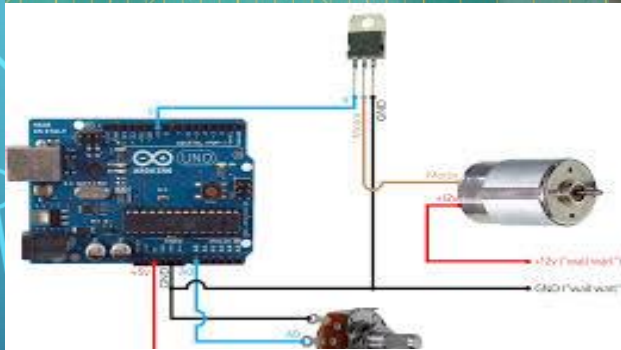
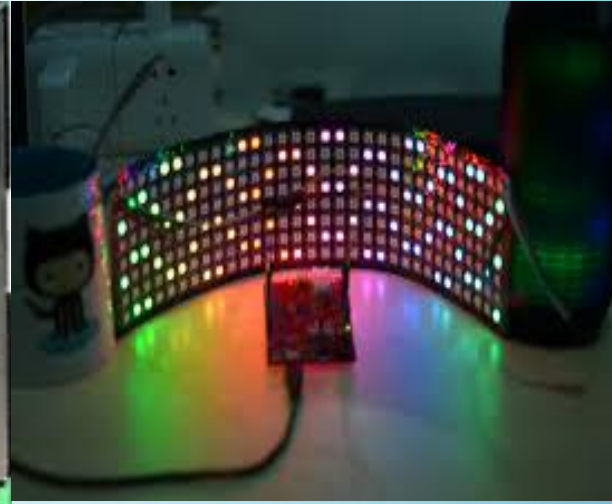
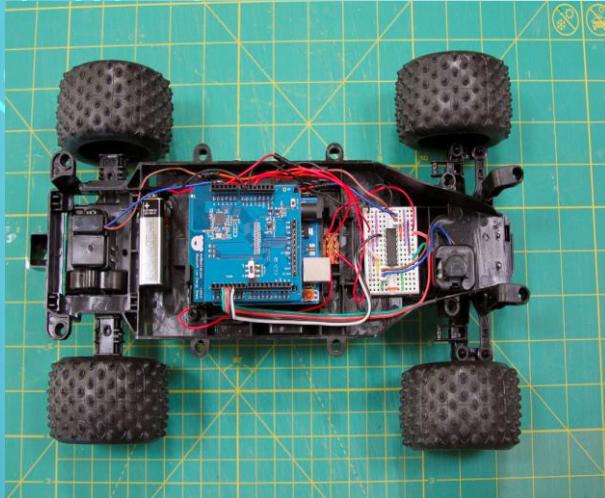


Stichting  
**DE JONGE ONDERZOEKERS  
AMERSFOORT**

Als techniek jouw hobby is!



# Dag1



Als techniek jouw hobby is!



# Programmeren

Globale variabelen declareren

Constanten declareren

```
void setup()
{
  ...
}
void loop()
{
  ...
}
```



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!





# Variabelen

int	geheel getal	-32,768 tot 32,767
long	geheel getal	-2,147,483,648 tot 2,147,483,647
float	reëel getal	+/-3.4028235E+38 (6-7 getallen nauwkeurigheid)
char	karakter	bijv: 'a', '-', '('
[ ]	array	bijv: int getallen[3]; int getallen[] = {1, 2, 3, 4};
"string"	char array	bijv: char s[20]; char s[] = "hallo"; // lengte = 6 !!



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!





# Constanten

Naam: Traditioneel met hoofdletters gescheiden door underscores '\_'.  
Bijv. uit de Arduino omgeving:

HIGH, LOW, INPUT, OUTPUT, INPUT\_PULLUP, LED\_BUILTIN

Zelf definiëren:

```
#define <naam> <waarde>
```

Bijv:

```
#define PIN_LED1 9
```

```
#define PIN_LED2 12
```

```
#define STR_LEN_MAX 20
```



# Functies

Belangrijke standaard functies:

`pinMode(pin, mode)`

`delay(ms)`

`millis()`

`digitalRead(pin)` // resultaat: HIGH of LOW

`digitalWrite(pin, waarde)`

`analogRead(pin)` // resultaat: 0 - 1023

`analogWrite(pin, waarde)` // waarde: 0 - 255, pin: 3, 5, 6, 9, 10, 11

`random()`



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!



# Functies

Definieer een blok code en geef het een naam zodat deze meerdere keren gebruikt kan worden.

```
void func1(int p1, int p2)
```

```
{
```

```
}
```

```
int func2(int p1)
```

```
{
```

```
    return p1 * 2;
```

```
}
```



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!



# If ... then ... else ...

Voer code uit als conditie 'cond' waar is en anders ...

```
if (cond)
```

```
{
```

```
}
```

```
else
```

```
{
```

```
}
```



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!





# If ... then ... else ...

Condities:

$x == y$  (x is equal to y)

$x != y$  (x is not equal to y)

$x < y$  (x is less than y)

$x > y$  (x is greater than y)

$x <= y$  (x is less than or equal to y)

$x >= y$  (x is greater than or equal to y)



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!



# If ... then ... else ...

```
if (cond1)
{ ... }
else if (cond2)
{ ... }
else if (cond3)
{ ... }
else
{ ... }
```



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!



# For loop

Herhaal de volgende code een aantal keer

```
for (i = 0; i < 10; i++) { ... }
```

ophogen

```
i++
```

```
i = i + 1
```

```
i++, j++
```

```
i = i + x
```

voortijdig afbreken: break

meteen volgende: continue



Stichting

**DE JONGE ONDERZOEKERS  
AMERSFOORT**

Als techniek jouw hobby is!



# While loop

Herhaal code zolang conditie 'cond' waar is

```
while (cond) { ... }
```

voortijdig afbreken: `break`

meteen volgende: `continue`



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!





# Commentaar

```
// Dit is commentaar op 1 regel
```

```
/* Dit is commentaar  
   op meerdere  
   regels
```

```
*/
```



Stichting  
**DE JONGE ONDERZOEKERS  
AMERSFOORT**

Als techniek jouw hobby is!



Dag 1

# Elektronica

- Weerstand
- LED (diode)
- Breadboard



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

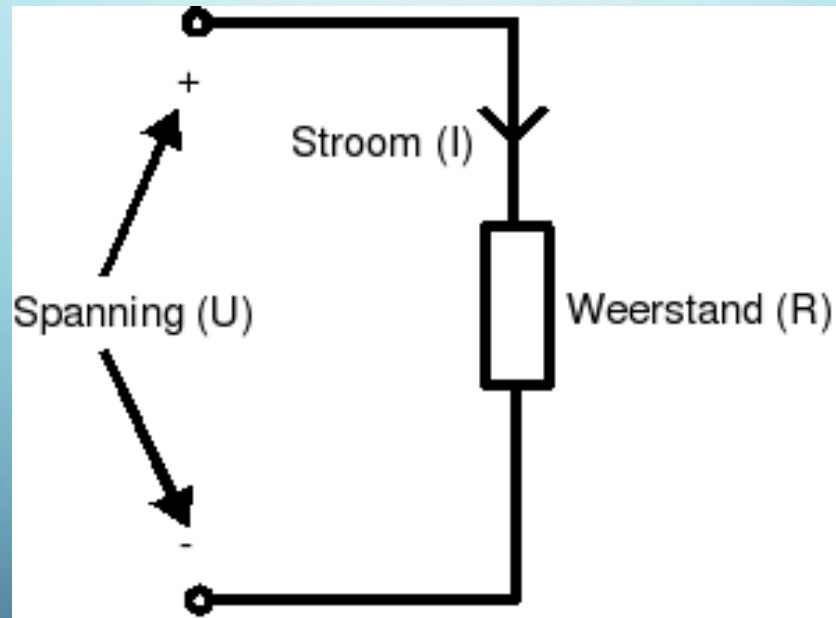
Als techniek jouw hobby is!



# Weerstand

- Spanning
- Stroom
- Weerstand

$$R = \frac{U}{I}$$



■ zwart 0	■ groen 5
■ bruin 1	■ blauw 6
■ rood 2	■ paars 7
■ oranje 3	■ grijs 8
■ geel 4	□ wit 9

ring 1 = cijfer  
ring 2 = cijfer  
ring 3 = aantal nullen



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

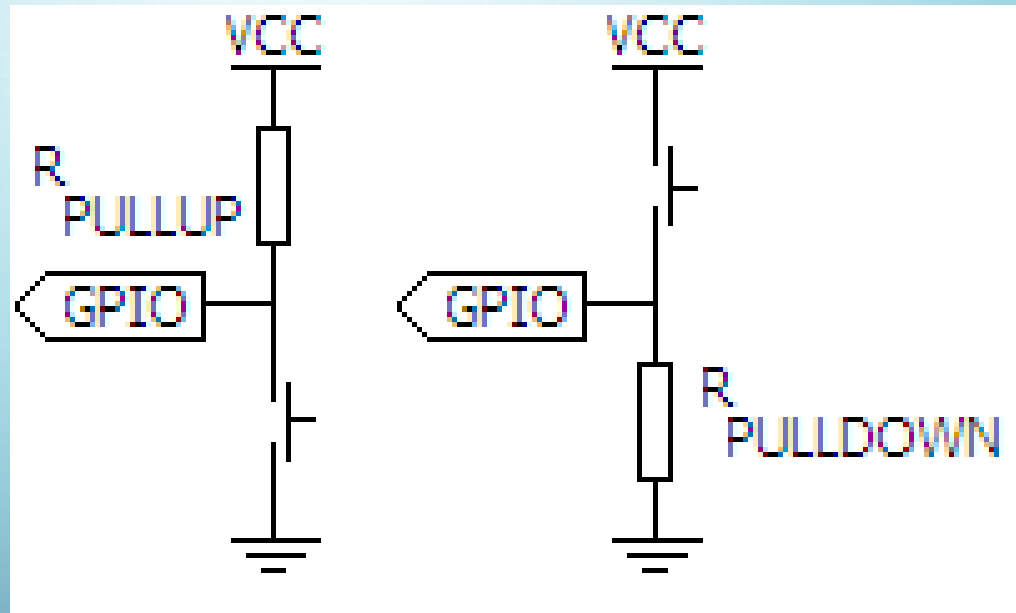
Als techniek jouw hobby is!



# Weerstand

•Pull-up

•Pull-down





# Weerstanden waarom hoog en laag

LED 470 Ohm -> datasheet

Digitaal 10.000 Ohm (10k Ohm) -> tussen 10k en 100k

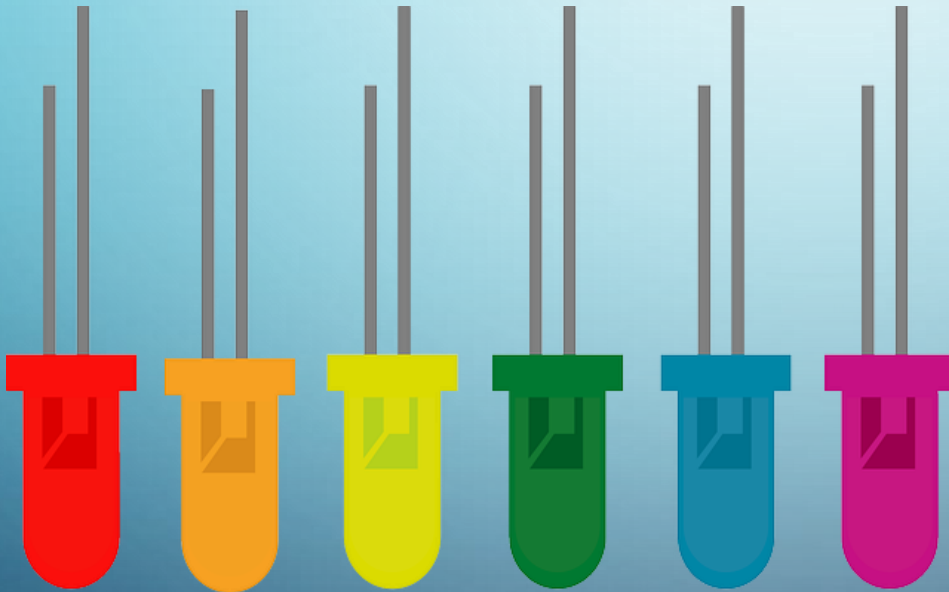
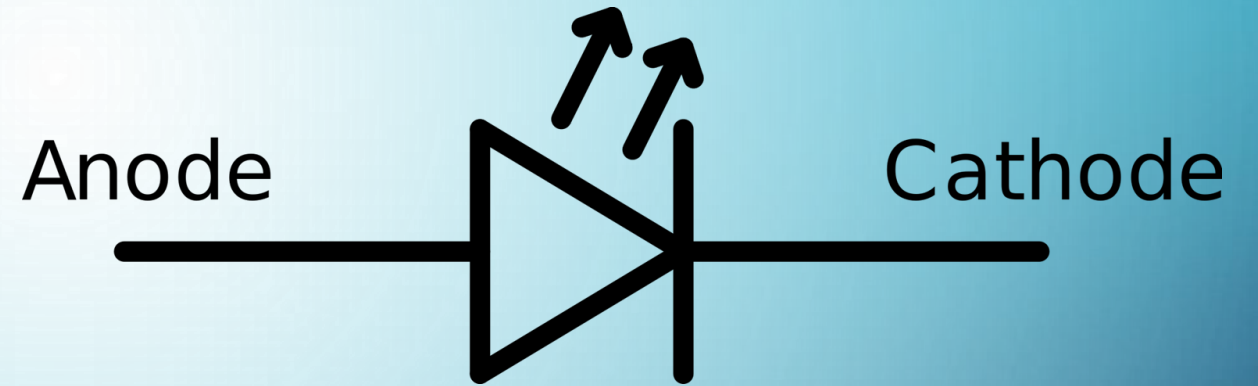
Temperatuur sensor 4700 Ohm (4k7) -> datasheet



Dag1

LED (diode)

•Éénrichtingsverkeer



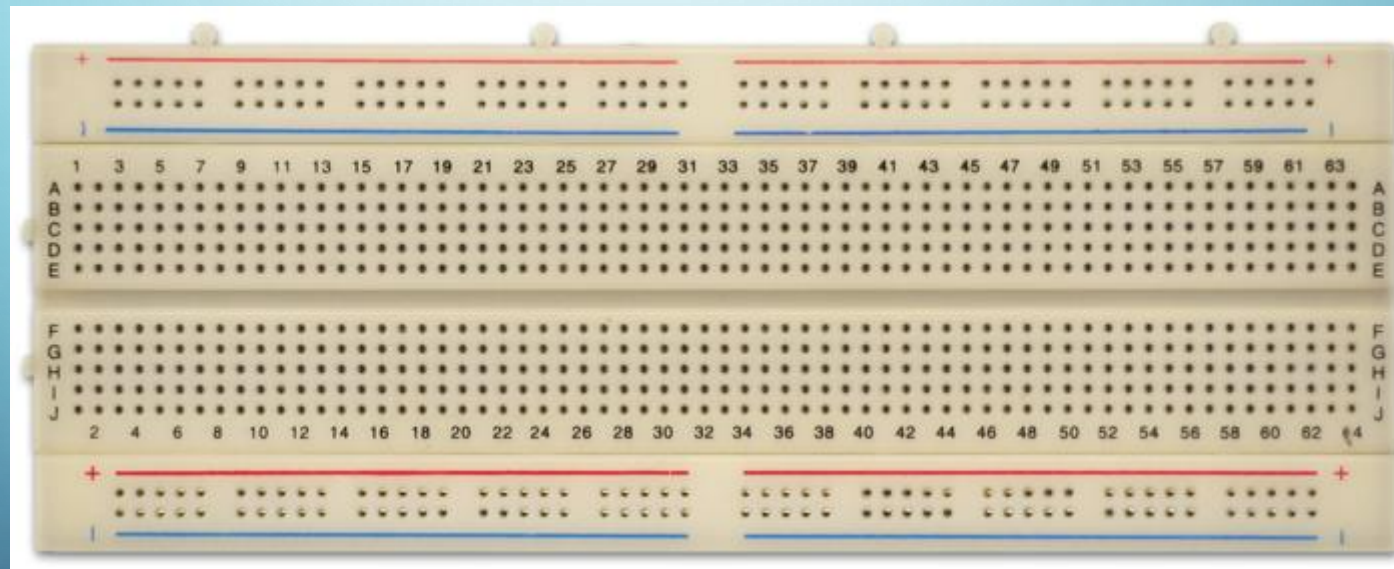
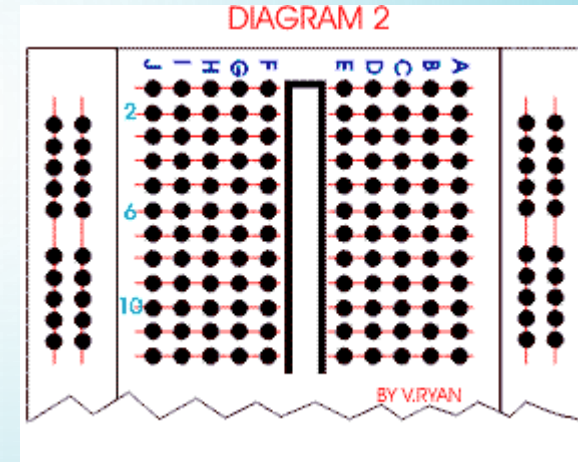
Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!



Dag1

# Breadboard



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!





Dag1

# Oefeningen



Stichting  
**DE JONGE ONDERZOEKERS  
AMERSFOORT**

Als techniek jouw hobby is!



Dag2

# Welkom!

Bij de arduino workshop voor beginners Dag 2



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**  
Als techniek jouw hobby is!



# Workshop Agenda

Dag 1 – Intro, theorie en kleine digitale projectjes

**Dag 2 – Theorie en Analoge projecten**

Dag 3 – Optional interfaces



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!





# Agenda Dag2

- Hoe liep dag 1
- Correcties?
- Vragen dag1?
- Functies
- PWM
- Oefeningen



# Hoe liep dag 1

- Huiswerk?
- Wie heeft wat vorige week gedaan
- Vragen n.a.v. Vorige week
- nieuwe ideeën



Dag2

Correcties?



Stichting  
**DE JONGE ONDERZOEKERS  
AMERSFOORT**

Als techniek jouw hobby is!





Dag2

vragen van dag1?



Stichting  
**DE JONGE ONDERZOEKERS  
AMERSFOORT**

Als techniek jouw hobby is!



# Builtin constants (arduino.h)

```
#define HIGH 0x1  
#define LOW 0x0
```

```
#define INPUT 0x0  
#define OUTPUT 0x1  
#define INPUT_PULLUP 0x2
```

```
#define PI 3.1415926535897932384626433832795  
#define HALF_PI 1.5707963267948966192313216916398  
#define TWO_PI 6.283185307179586476925286766559  
#define DEG_TO_RAD 0.017453292519943295769236907684886  
#define RAD_TO_DEG 57.295779513082320876798154814105  
#define EULER 2.718281828459045235360287471352
```

```
#define SERIAL 0x0  
#define DISPLAY 0x1
```

```
#define LSBFIRST 0  
#define MSBFIRST 1
```



# Builtin constants (binary.h)

```
#define B0 0
#define B00 0
#define B000 0
#define B0000 0
#define B00000 0
#define B000000 0
#define B0000000 0
#define B00000000 0
#define B1 1
#define B01 1
#define B001 1
#define B0001 1
#define B00001 1
#define B000001 1
#define B0000001 1
#define B00000001 1
#define B10 2
#define B010 2
#define B0010 2
#define B00010 2
#define B000010 2
#define B0000010 2
#define B00000010 2
```





# ARDUINO CHEAT SHEET V.02B

Mostly taken from the extended reference:  
<http://arduino.cc/en/Reference/Extended>  
 Gavin Smith – Robots and Dinosaurs, The Sydney Hackspace



**Structure**  
 void setup() void loop()

## Control Structures

```
if (x<5){ } else { }
switch (myvar) {
  case 1:
    break;
  case 2:
    break;
  default:
}
for (int i=0; i <= 255; i++){ }
while (x<5){ }
do { } while (x<5);
continue; //Go to next in do/for/while loop
return x; // Or 'return;' for voids.
goto // considered harmful :-)
```

## Further Syntax

```
// (single line comment)
/* (multi-line comment) */
#define DOZEN 12 //Not baker's!
#include <avr/pgmspace.h>
```

## General Operators

```
= (assignment operator)
+ (addition) - (subtraction)
* (multiplication) / (division)
% (modulo)
== (equal to) != (not equal to)
< (less than) > (greater than)
<= (less than or equal to)
>= (greater than or equal to)
&& (and) || (or) ! (not)
```

## Pointer Access

```
& reference operator
* dereference operator
```

## Bitwise Operators

```
& (bitwise and) | (bitwise or)
^ (bitwise xor) ~ (bitwise not)
<< (bitshift left) >> (bitshift right)
```

## Compound Operators

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

## Constants

```
HIGH | LOW
INPUT | OUTPUT
true | false
143 // Decimal number
0173 // Octal number
B11011111 // Binary (8-bits only)
0x7B // Hex number
7U // Force unsigned
10L // Force long
15UL // Force long unsigned
10.0 // Forces floating point
2.4e5 // 245,000
```

## Data Types

```
boolean (0, 1, false, true)
char (e.g. 'a' -128 to 127)
unsigned char (0 to 255)
byte (0 to 255)
int (-32,768 to 32,767)
unsigned int (0 to 65535)
word (0 to 65535)
long (-2,147,483,648 to 2,147,483,647)
unsigned long (0 to 4,294,967,295)
float (-3.4028235E+38 to 3.4028235E+38)
double (currently same as float)
sizeof(myint) // returns 2 bytes
```

## Strings

```
char S1[15];
char S2[8]={'a','r','d','u','i','n','o'};
char S3[8]={'a','r','d','u','i','n','o','\0'};
//Included \0 null termination
char S4[]="arduino";
char S5[8]="arduino";
char S6[15]="arduino";
```

## Arrays

```
int myInts[6];
int myPins[] = {2, 4, 8, 3, 6};
int mySensVals[6] = {2, 4, -8, 3, 2};
```

## Conversion

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

## Qualifiers

```
static // persists between calls
volatile // use RAM (nice for ISR)
const // make read-only
PROGMEM // use flash
```

## Digital I/O

```
pinMode(pin, [INPUT,OUTPUT])
digitalWrite(pin, value)
int digitalRead(pin)
//Write High to inputs to use pull-up res
```

## Analog I/O

```
analogReference([DEFAULT,INTERNAL,EXTERNAL])
int analogRead(pin) //Call twice if switching pins from high Z source.
analogWrite(pin, value) // PWM
```

## Advanced I/O

```
tone(pin, freqhz)
tone(pin, freqhz, duration_ms)
noTone(pin)
shiftOut(dataPin, clockPin, [MSBFIRST,LSBFIRST], value)
unsigned long pulseIn(pin, [HIGH,LOW])
```

## Time

```
unsigned long millis() // 50 days overflow.
unsigned long micros() // 70 min overflow
delay(ms)
delayMicroseconds(us)
```

## Math

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

## Random Numbers

```
randomSeed(seed) // Long or int
long random(max)
long random(min, max)
```

## Bits and Bytes

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

## External Interrupts

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

## Libraries:

```
Serial.
  begin([300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600, 115200])
  end()
  int available()
  int read()
  flush()
  print()
  println()
  write()
```

```
EEPROM (#include <EEPROM.h>)
  byte read(intAddr)
  write(intAddr,myByte)
```

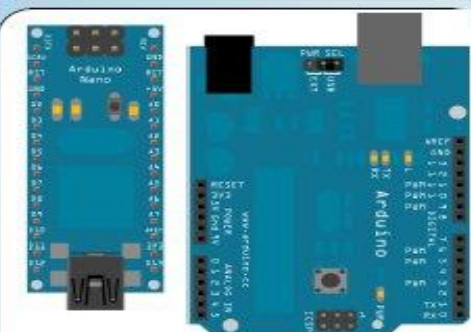
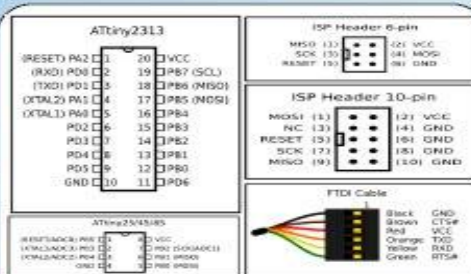
```
Servo (#include <Servo.h>)
  attach(pin, [min_uS, max_uS])
  write(angle) // 0-180
  writeMicroseconds(uS) //1000-2000, 1500 is midpoint
  read() // 0-180
  attached() //Returns boolean
  detach()
```

```
SoftwareSerial(RxPin,TxPin)
// #include<SoftwareSerial.h>
begin(longSpeed) // up to 9600
char read() // blocks till data
print(myData) or println(myData)
```

```
Wire (#include <Wire.h>) // For I2C
begin() // Join as master
begin(addr) // Join as slave @ addr
requestFrom(address, count)
beginTransmission(addr) // Step 1
send(mybyte) // Step 2
send(char * mystring)
send(byte * data, size)
endTransmission() // Step 3
byte available() // Num of bytes
byte receive() //Return next byte
onReceive(handler)
onRequest(handler)
```

	ATmega168	ATmega328	ATmega1280
Flash (2K for bootloader)	16KB	32KB	128KB
SRAM	1KB	2KB	8KB
EEPROM	512B	1KB	4KB

	Duemilanove/ Nano/ Pro/ ProMini	Mega
# of IO	14 + 6 analog (Nano has 14+8)	54 + 16 analog
Serial Pins	0 - RX 1 - TX	0 - RX1 1 - TX1 19 - RX2 18 - TX2 17 - RX3 16 - TX3 15 - RX4 14 - TX4
Ext interrupts	2 - (Int 0) 3 - (Int 1)	2,3,21,20,19,18 (IRQ0-IRQ5)
PWM pins	5,6 - Timer 0 9,10 - Timer 1 3,11 - Timer 2	0-13 53 - SS 51 - MOSI 50 - MISO 52 - SCK 20 - SDA 21 - SCL
SPI	10 - SS 11 - MOSI 12 - MISO 13 - SCK	
I2C	Analogs - SDA Analogs - SCL	



Pics from Fritzing.Org under C.C. license

# Functies

Definieer een blok code en geef het een naam zodat deze meerdere keren gebruikt kan worden.

```
void func1(int p1, int p2)
```

```
{  
}
```

```
int func2(int p1)
```

```
{  
    return p1 * 2;  
}
```





## Funcities - gebruik

```
int getal;  
getal = gemiddelde(5,6,8);  
.....  
getal = gemiddelde(1,4,5);  
.....  
getal = gemiddelde(9,3,3);
```





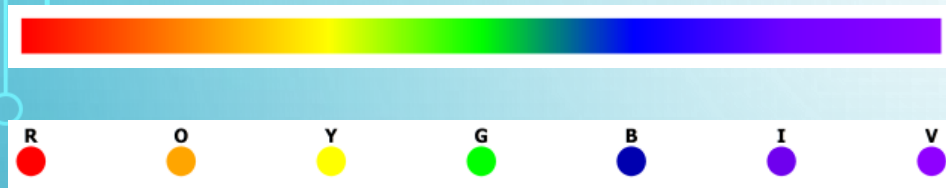
## Funcities - gebruik

```
int getal;  
getal = gemiddelde(5,6,8);  
.....  
getal = gemiddelde(1,4,5);  
.....  
getal = gemiddelde(9,3,3);
```

```
int function gemiddelde(int a,int b,int c)  
{ int resultaat = (a+b+c)/3;  
  return resultaat;  
}
```



# PWM - RGB mengen



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

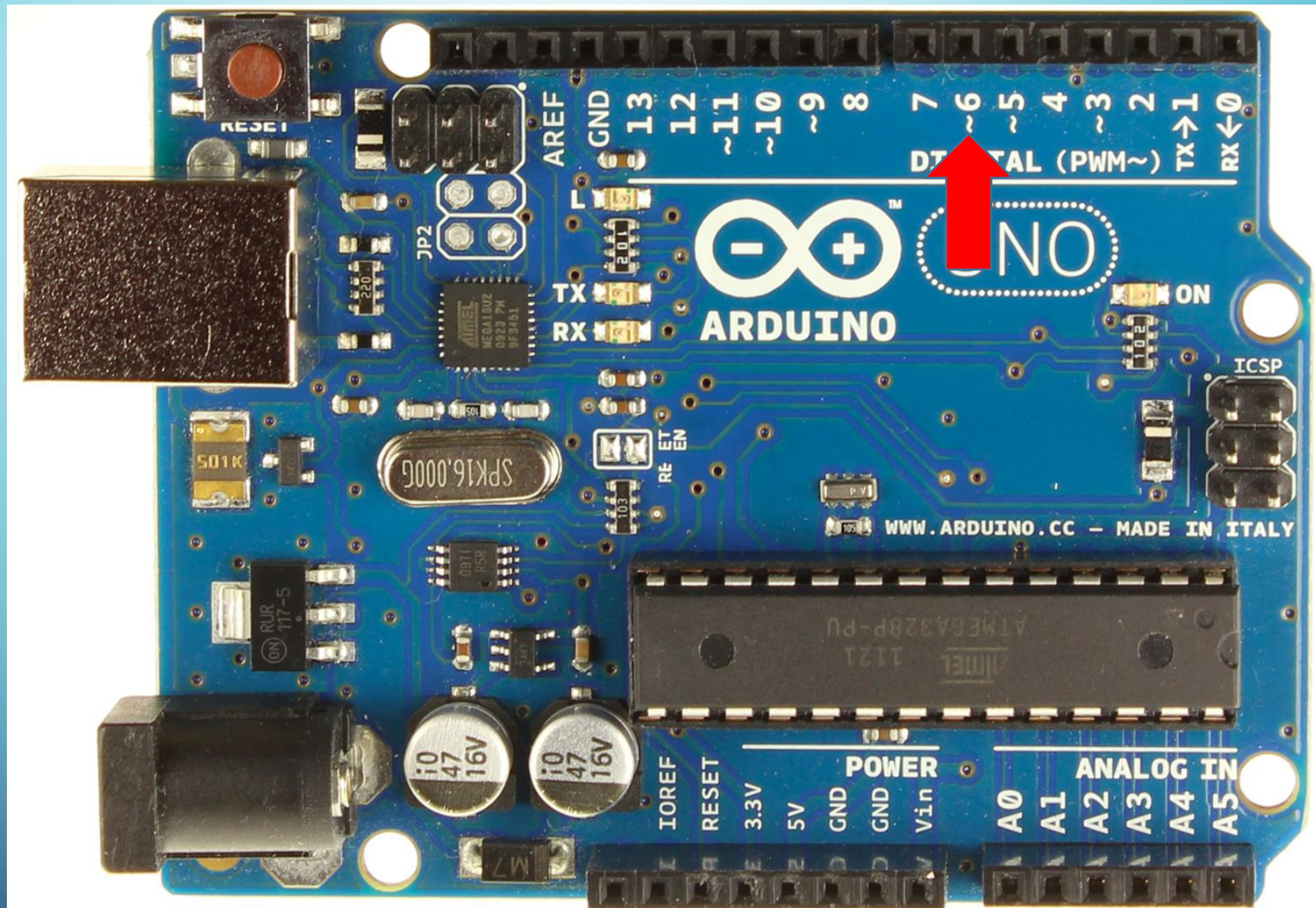
Als techniek jouw hobby is!





Dag2

PWM



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

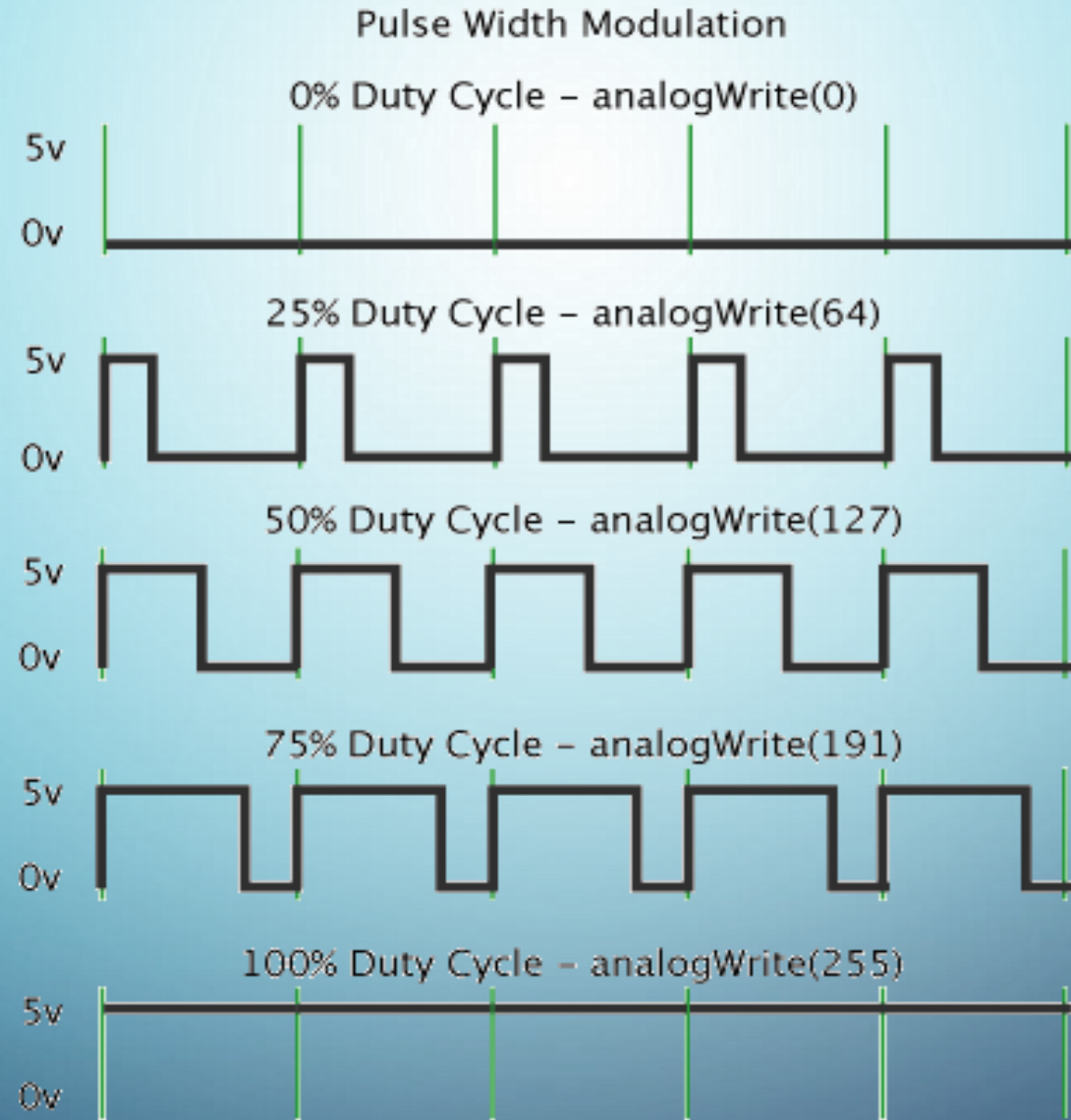
Als techniek jouw hobby is!





# PWM

(Super blink)



Dag2

# Oefeningen



Stichting  
**DE JONGE ONDERZOEKERS  
AMERSFOORT**

Als techniek jouw hobby is!



Dag 3

# Welkom!

Bij de arduino workshop voor beginners Dag 3



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!





Dag 3

# Workshop Agenda

Dag 1 – Intro, theorie en kleine digitale projectjes

Dag 2 – Theorie en Analoge projecten

**Dag 3 – Optional interfaces**



Stichting  
**DE JONGE ONDERZOEKERS  
AMERSFOORT**

Als techniek jouw hobby is!



# Agenda Dag 3

- Hoe liep dag 2
- Correcties?
- Vragen?
- herhaling dag 2
- I2C/SPI/OneWire/Serial
- Oefeningen



Dag 3

Correcties?



Stichting  
**DE JONGE ONDERZOEKERS  
AMERSFOORT**

Als techniek jouw hobby is!





Dag 3

vragen van dag 2?

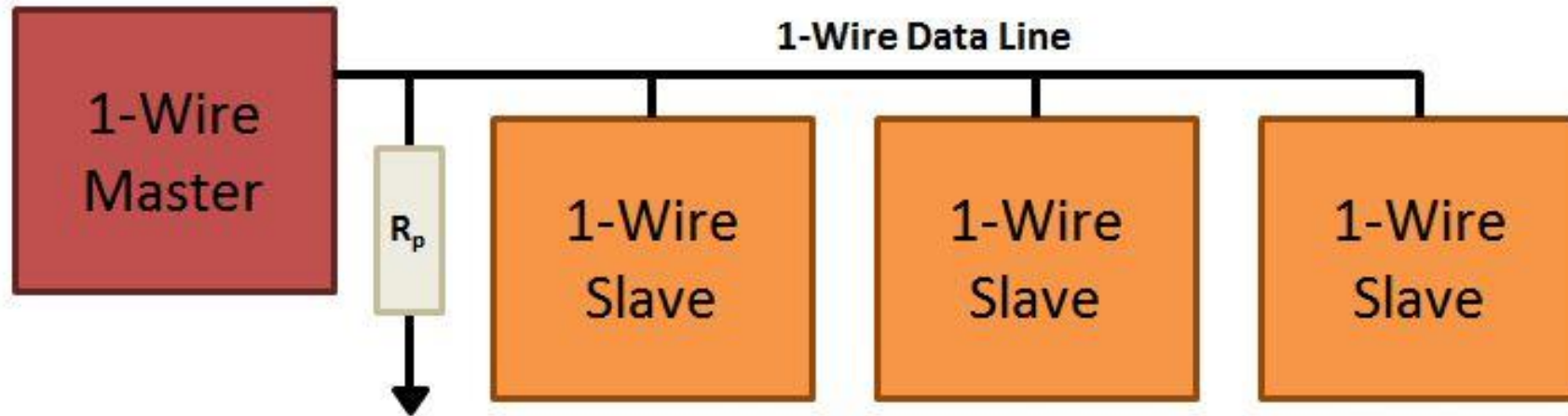


Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!



## I2C/SPI/OneWire/Serial



- **1-Wire Data line** is the master data output, slave data input, and it carries data from the master to the slave.
- $R_p$  are 4.7K pull-up resistor
- Slave address by unique 64-bit code, consisting of an 8-bit family code, a 48-bit serial number, and an 8-bit CRC.



Dag 3

# Oefeningen



Stichting  
**DE JONGE ONDERZOEKERS  
AMERSFOORT**

Als techniek jouw hobby is!





Dag 3

## Afsluiting

- Hoe liep dag 3?
- Wat na dag 3?



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!



Dag 3

# Extra oefeningen

Communiceren, bijv. NRF24L01

Motoren etc.



Stichting  
**DE JONGE ONDERZOEKERS**  
**AMERSFOORT**

Als techniek jouw hobby is!

