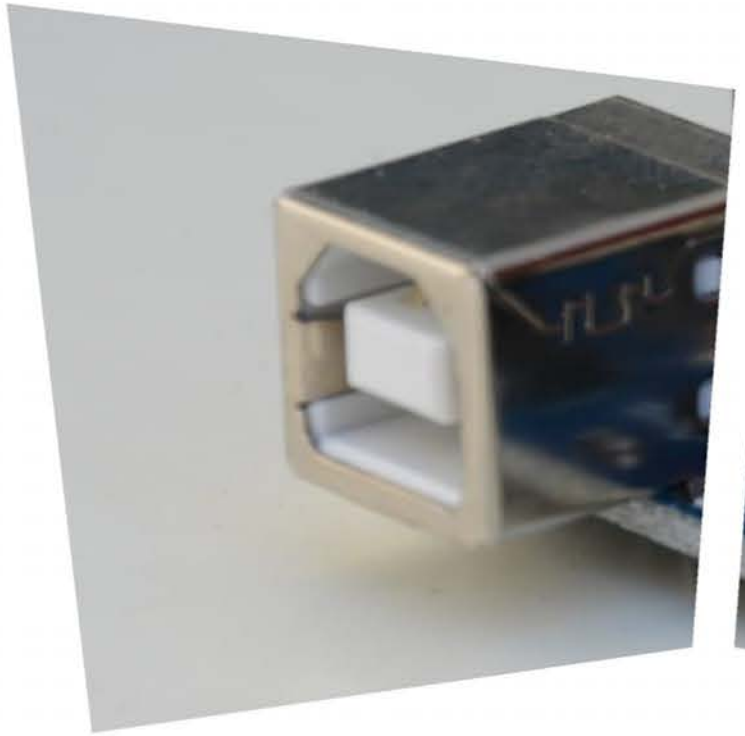
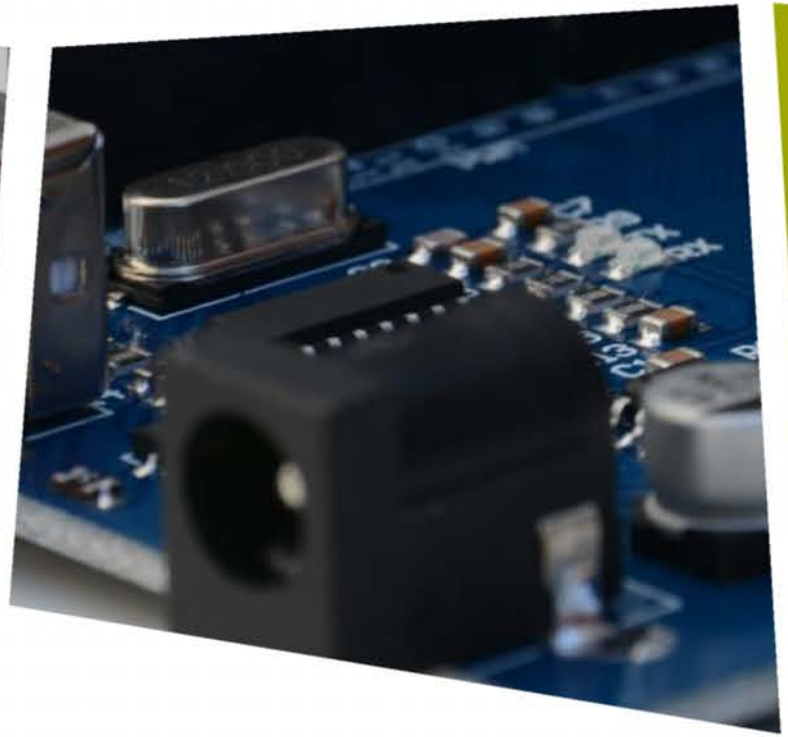




2023

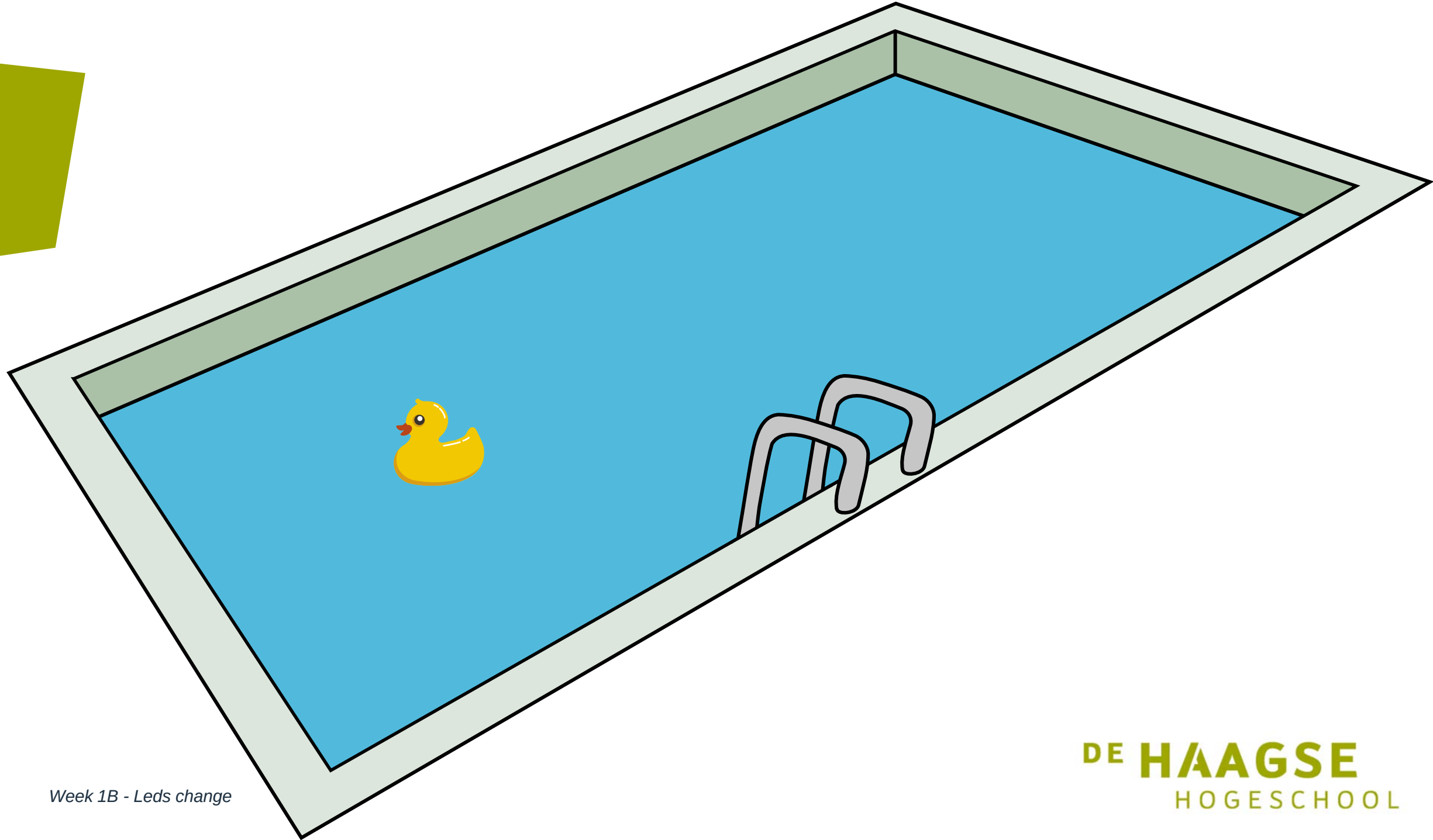
G.M. Tuk
X. van Rijnsoever

LEDS CHANGE

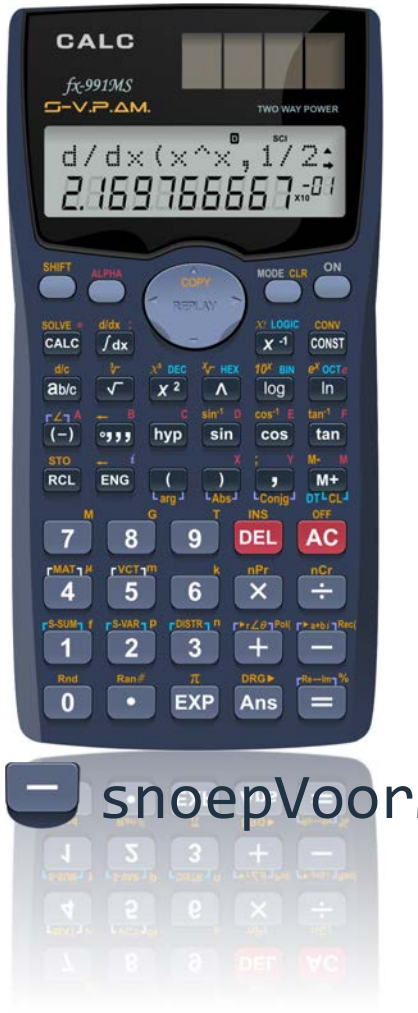
Microcontroller Programmeren 1 – Week 1B

DE HAAGSE HOGESCHOOL


leds change
YOU. US. THE WORLD.



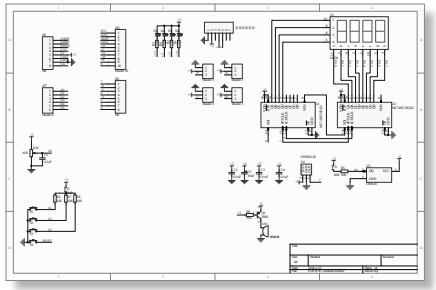
```
// Invoer
printf("Hoeveel studenten zijn er?\n");
scanf("%d", &aantalDelers);
aantalDelers = aantalDelers + 1;

// Verwerking
int aantalSnoep = aantalZakken * snoepInZak;
int snoepPerPersoon = aantalSnoep / aantalDelers - snoepVoorMa;

// Uitvoer
printf("Snoepjes per persoon: %d\n", snoepPerPersoon);
printf("Snoepjes over: %d\n", aantalSnoep % aantalDelers);
```

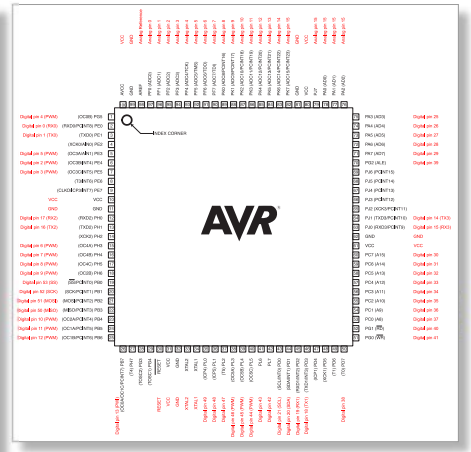


DE HAAGSE HOGESCHOOL

Wat gaan we doen vandaag?

- Kleine aanvulling: beslissingen met **if**
- Programmeren AVR microcontroller: knipperend ledje
  - Testen toolchain
  - Leren hoe je met input en output van een microcontroller werkt
  - Werken met **elektrisch schema**, **datasheet** en **pinouts**





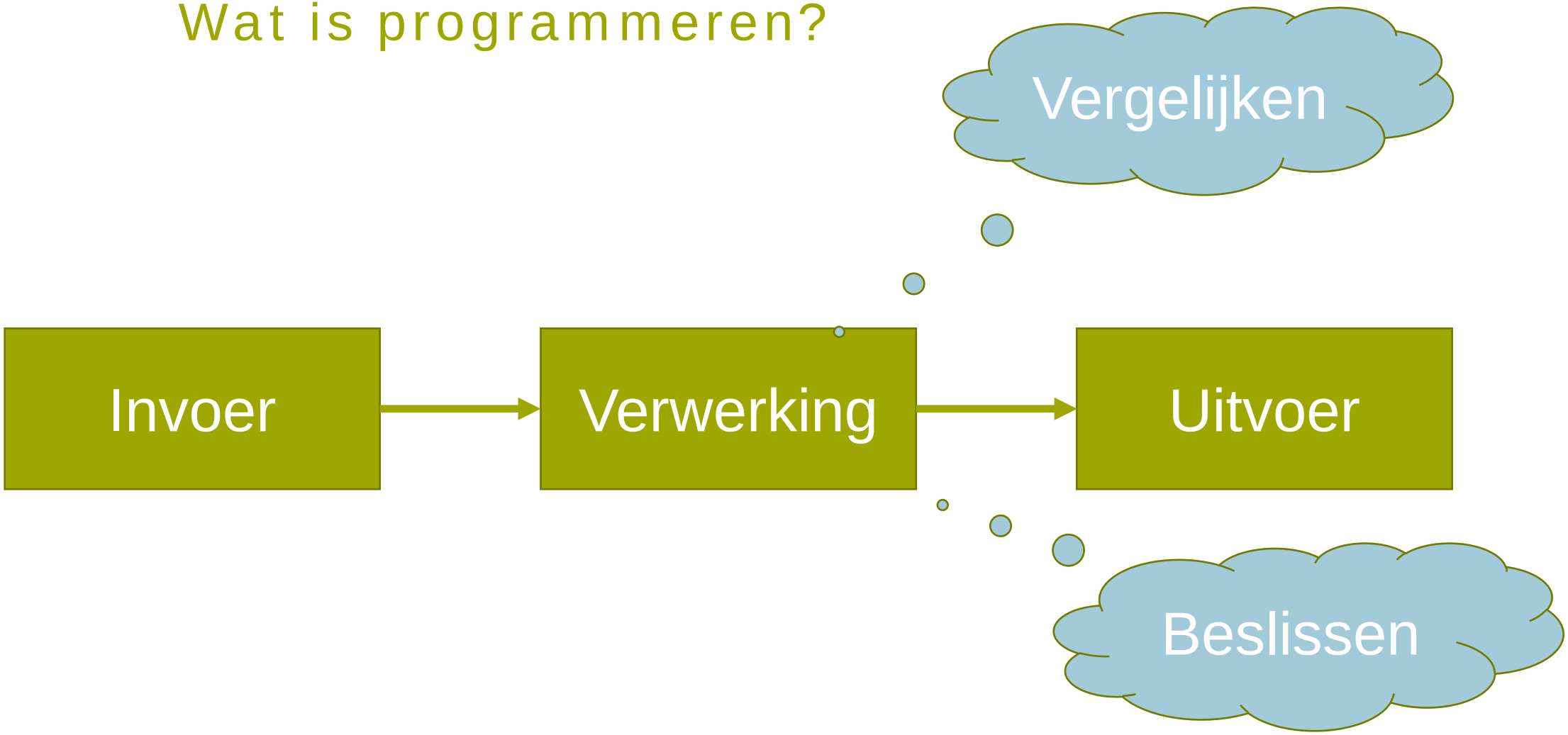
HAAGSE HOGESCHOOL

Commentaar

```
// Beginwaarden van de variabelen
int invoer = 20;
int uitkomst = 0;

/* De berekening van de resultaten vindt
   plaats aan het begin omdat ze al bij de
   controle van de invoer nodig zijn. */
resultaat1 = 2 * invoer + marge;
resultaat2 = invoer + uitkomst;
```

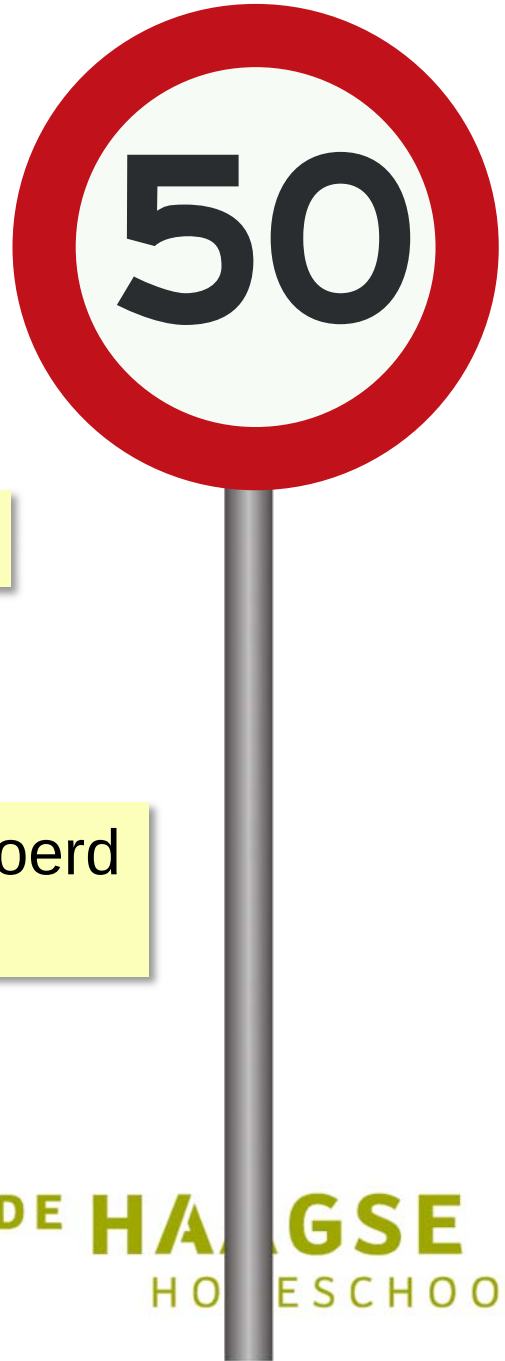
```c
int main(void)
{
	int waarde;
	scanf("%d", &waarde);

	// Print melding als waarde groter is dan 1000

	return 0;
}
```

Wat is programmeren?



DE HAAGSE HOGESCHOOL

```c
int main(void)
{
    int snelheid;
    scanf("%d", &snelheid);
    if (snelheid > 50)
    {
        printf("U rijdt te hard.\n");
    }
    return 0;
}
```

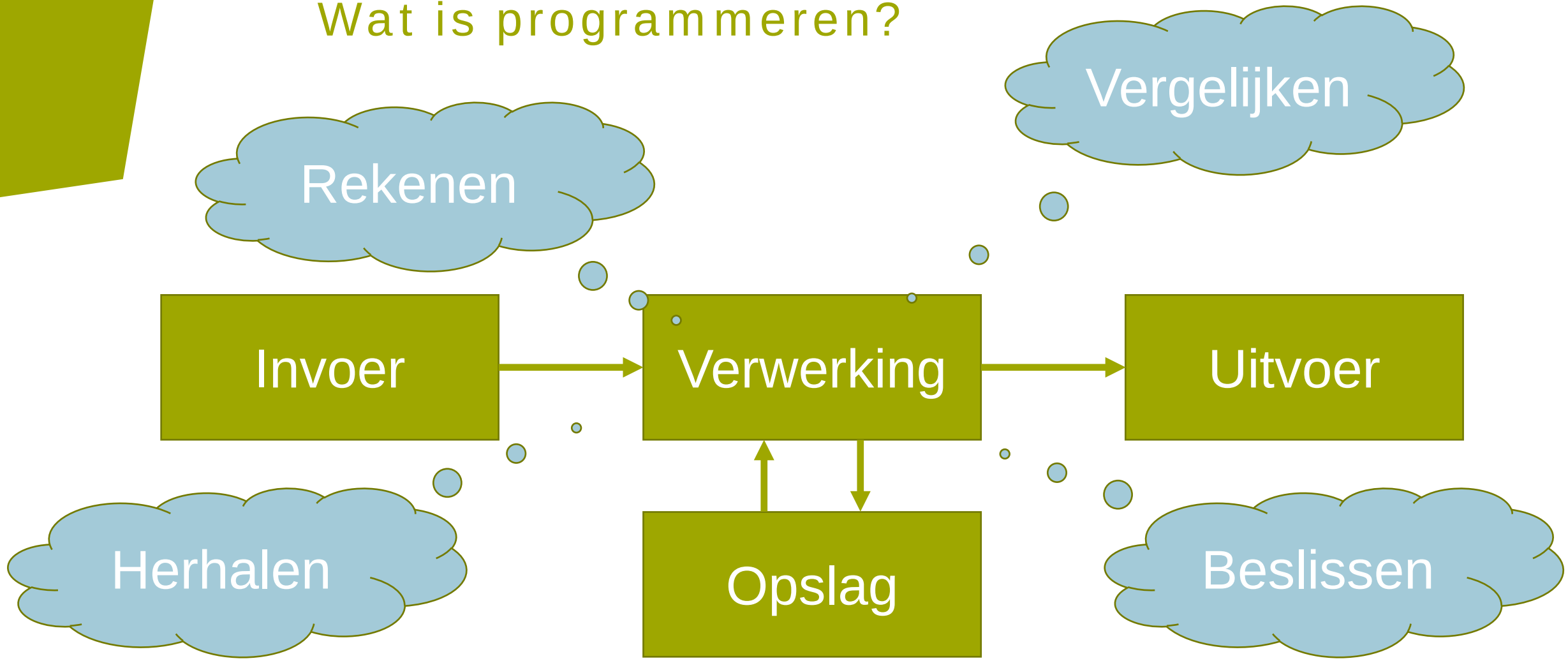


conditie: groter dan

body: code in { } wordt uitgevoerd als conditie waar is

DE HAAGSE HOGESCHOOL

Wat is programmeren?



DE HAAGSE HOGESCHOOL

Wat is programmeren?

Vergelijken

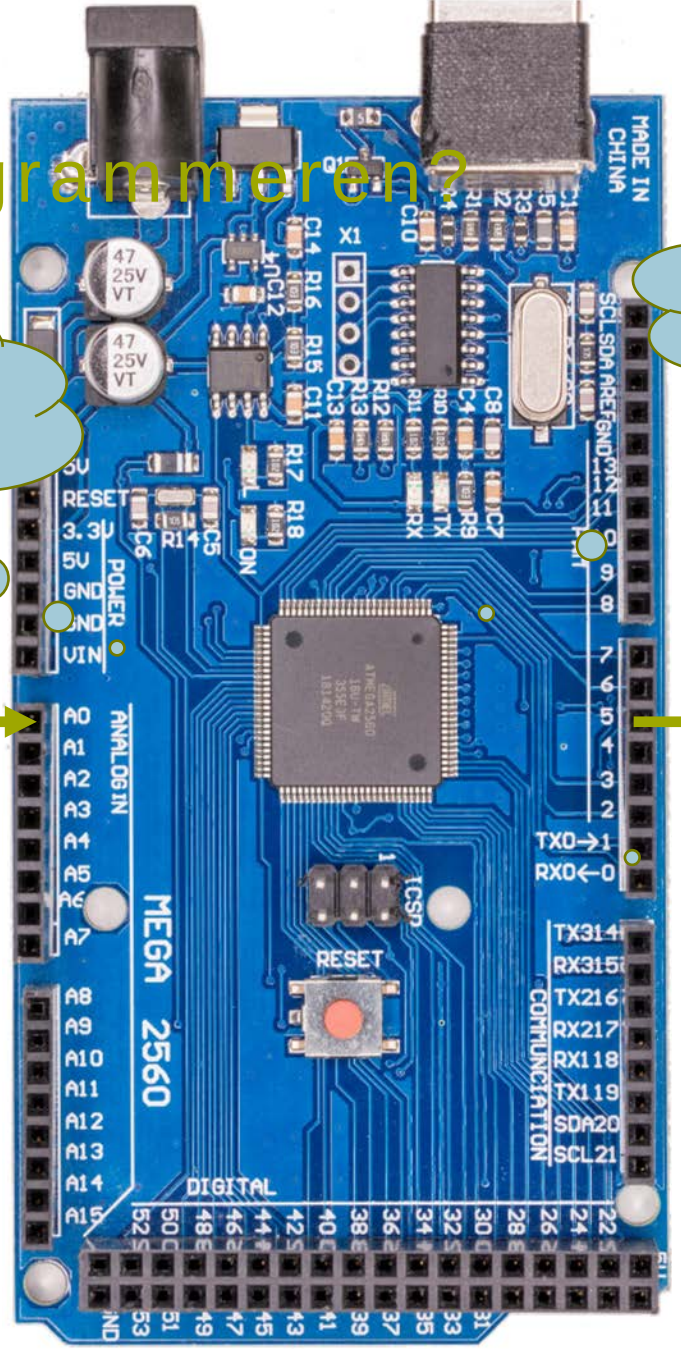
Rekenen

Invoer

Uitvoer

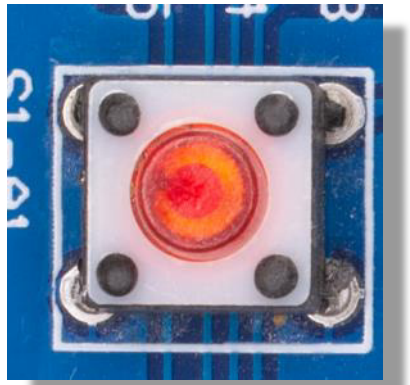
Herhalen

Beslissen



DE HAAGSE HOGESCHOOL

Wat is programmeren?

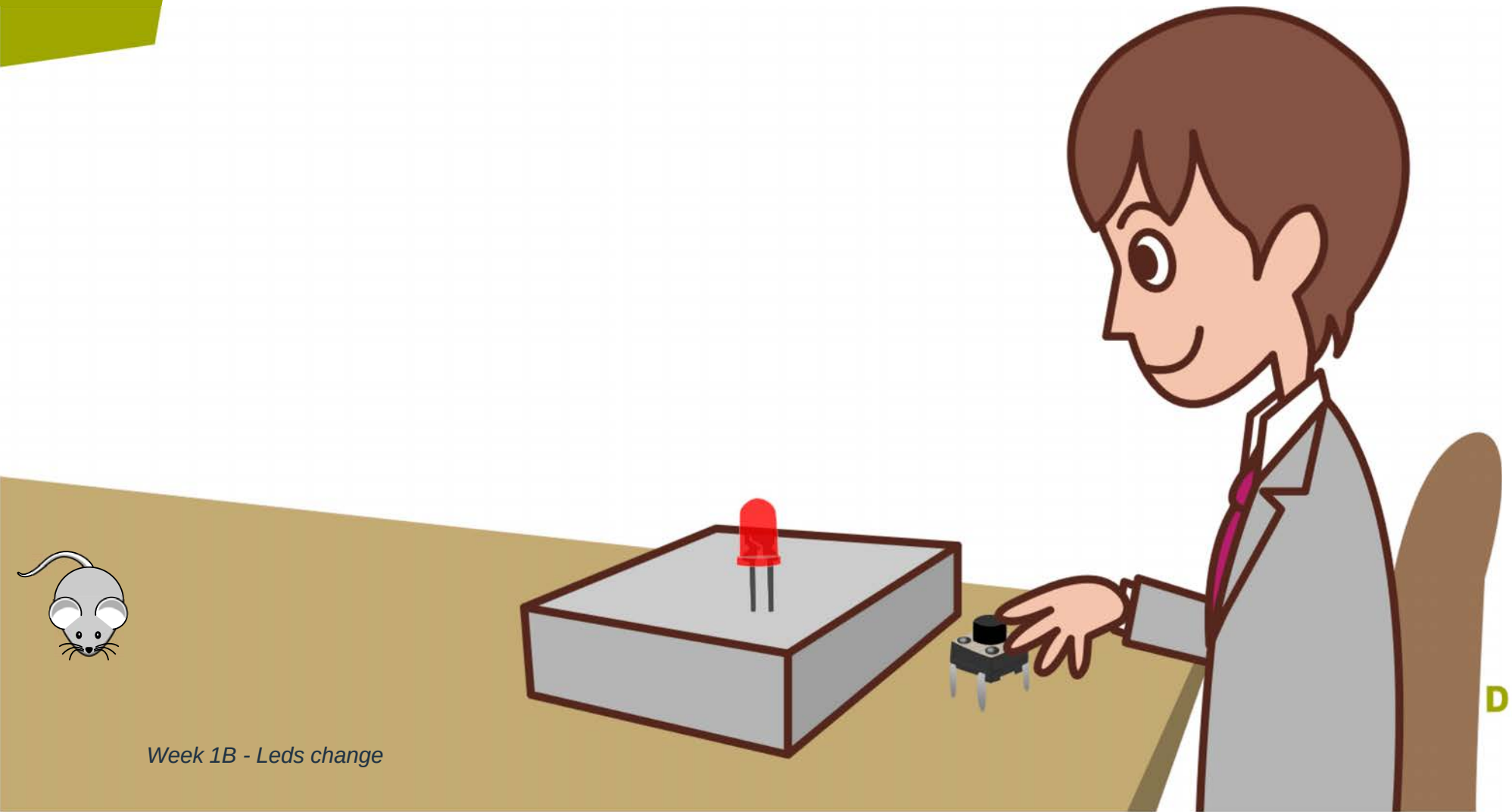


Invoer → Verwerking → Uitvoer

foto's: Gerard Tuk

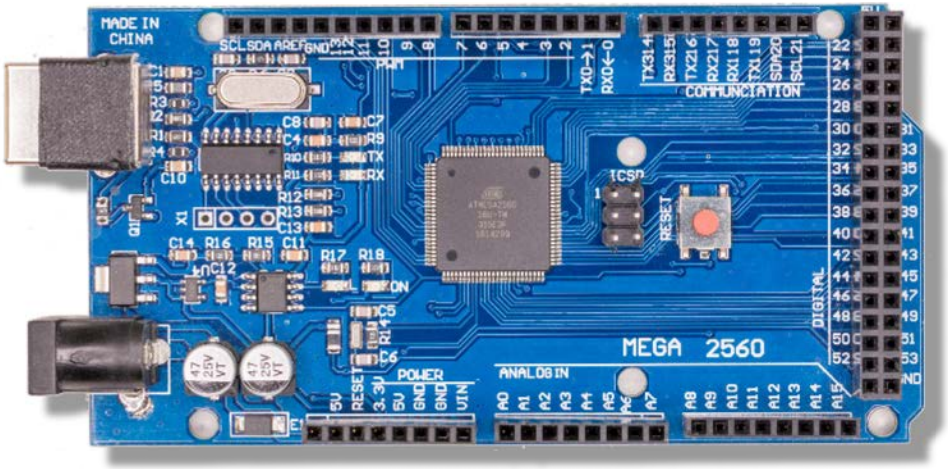
Afbeelding van OpenClipart-Vectors via Pixabay

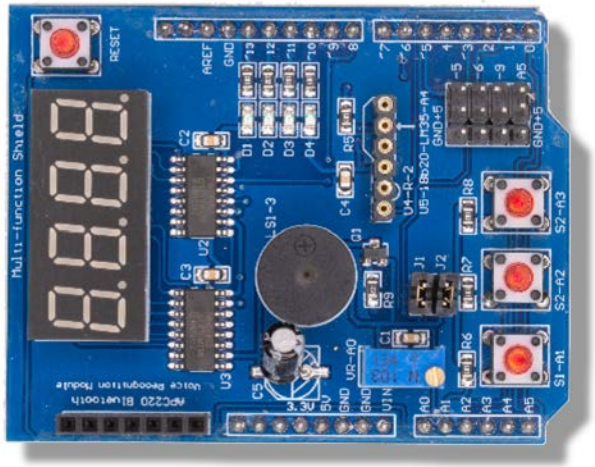
Invoer en uitvoer...



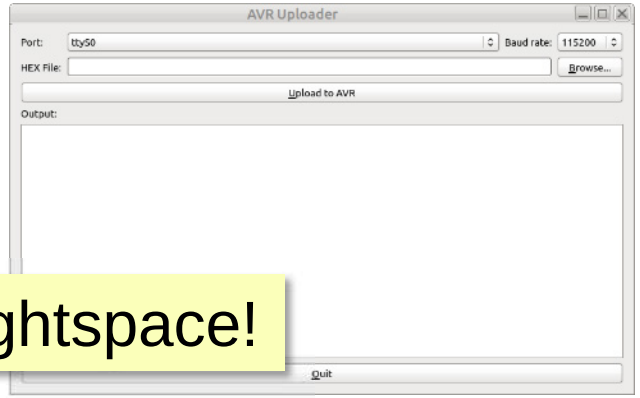
DE HAAGSE HOGESCHOOL

Benodigdheden





AVR GCC
12.1.0



Zie installatiehandleiding op Brightspace!

DE HAAGSE HOGESCHOOL

Knipperend ledje

Benodigde kennis hardware kant:

- Elektrisch schema shield
  - Hoe is het ledje aangesloten?
  - Op welke Arduino-pin is het ledje aangesloten?
- Pinout Arduino
  - Op welke AVR pin zit de Arduino pin aangesloten?

Knipperend ledje

Stappen software:

- Configureer juiste pin als output pin
- Ledje aan- en uitschakelen
- Vastgestelde tijd kunnen wachten

Knipperend ledje

```
#include <avr/io.h>

int main(void)
{
    // configureer pin
    // zet led aan
    // wacht 500 ms
    // zet led uit
    // wacht 500 ms
    return 0;
}
```

DE HAAGSE HOGESCHOOL

Herhalen met while lus

```
while (1)
{
      // code die telkens wordt herhaald
      ...
}
```

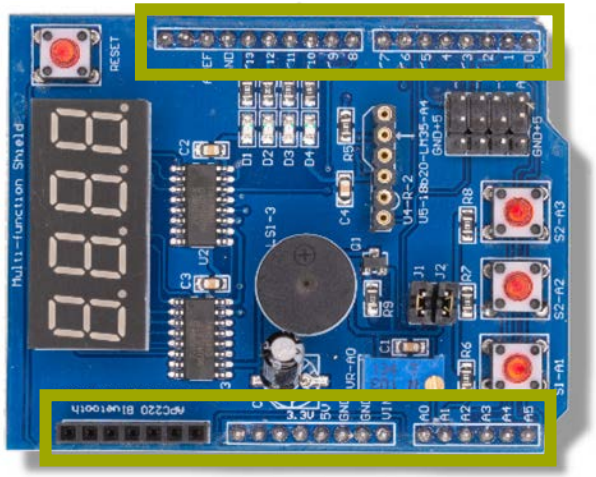
Let op dat je hierachter geen puntkomma zet!

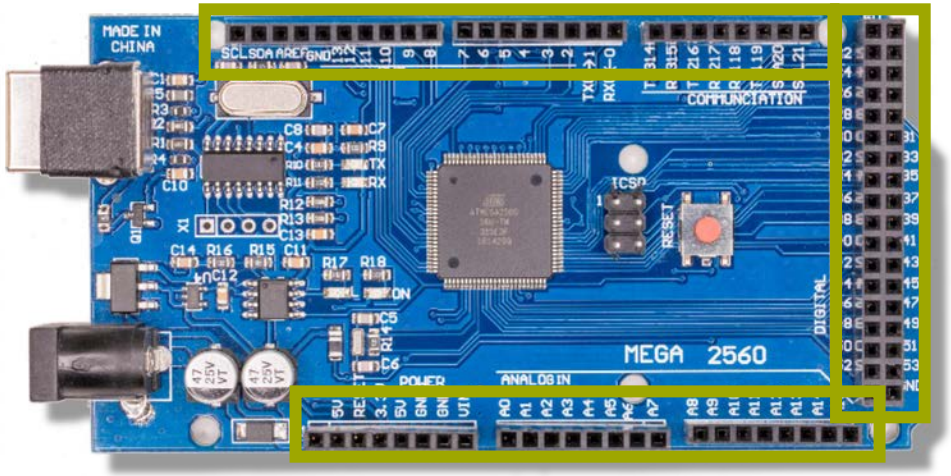
```c
#include <avr/io.h>

int main(void)
{
	// configureer pin
	while (1)
	{
		// zet led aan
		// wacht 500 ms
		// zet led uit
		// wacht 500 ms
	}
	return 0;
}
```

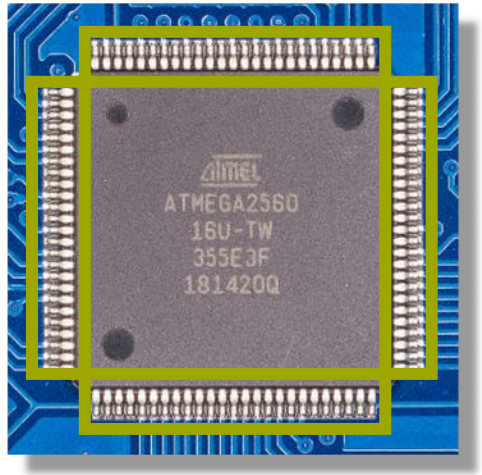

“Door de bo en het bos niet meer zien”

Er zijn maar drie soorten bomen

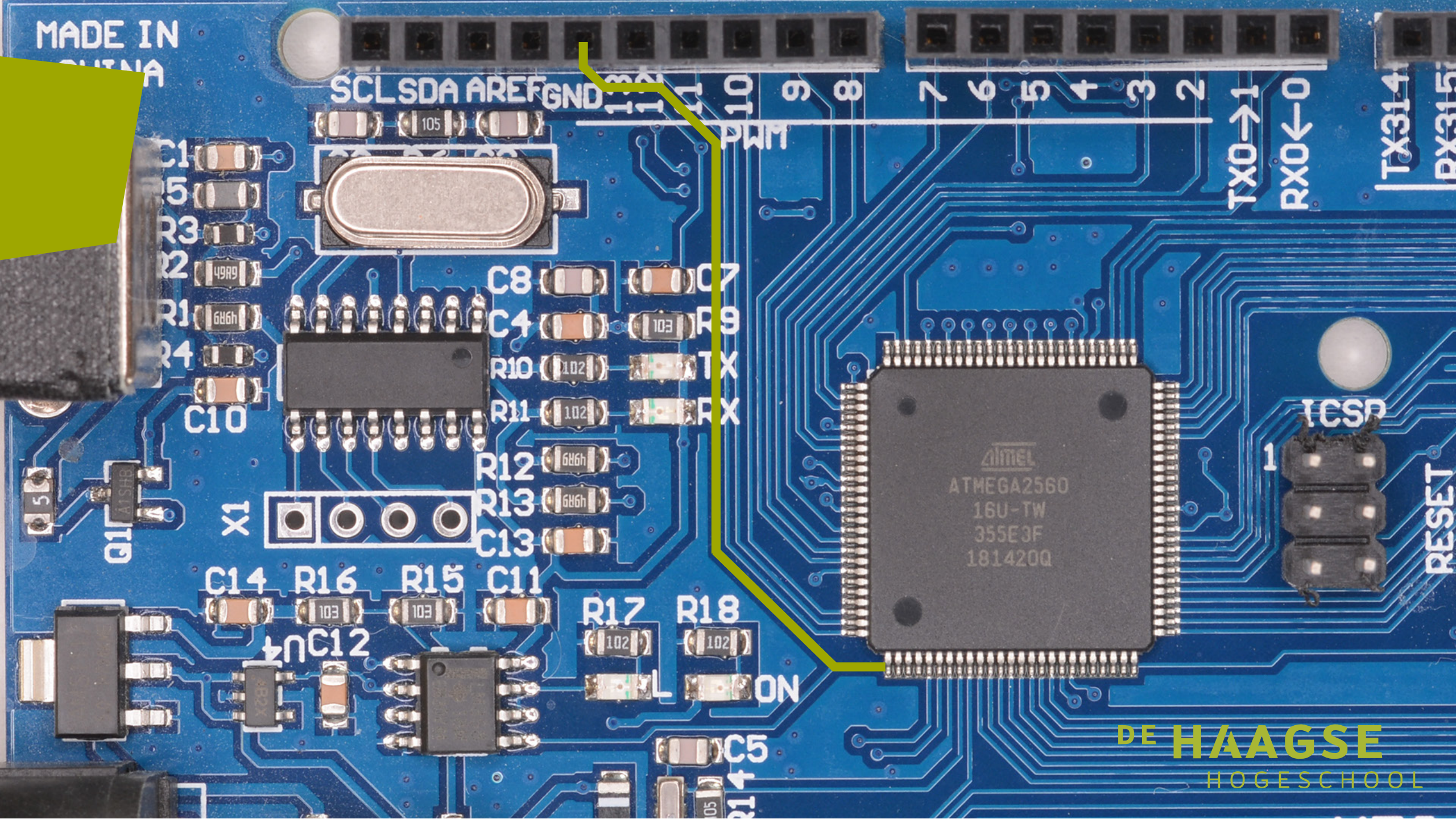
- De Atmel ATmega2560 Microcontroller
- Het Arduino bordje (of de kloon, die lekker goedkoop is)
- Het Velleman multi-function shield (of de kloon...)

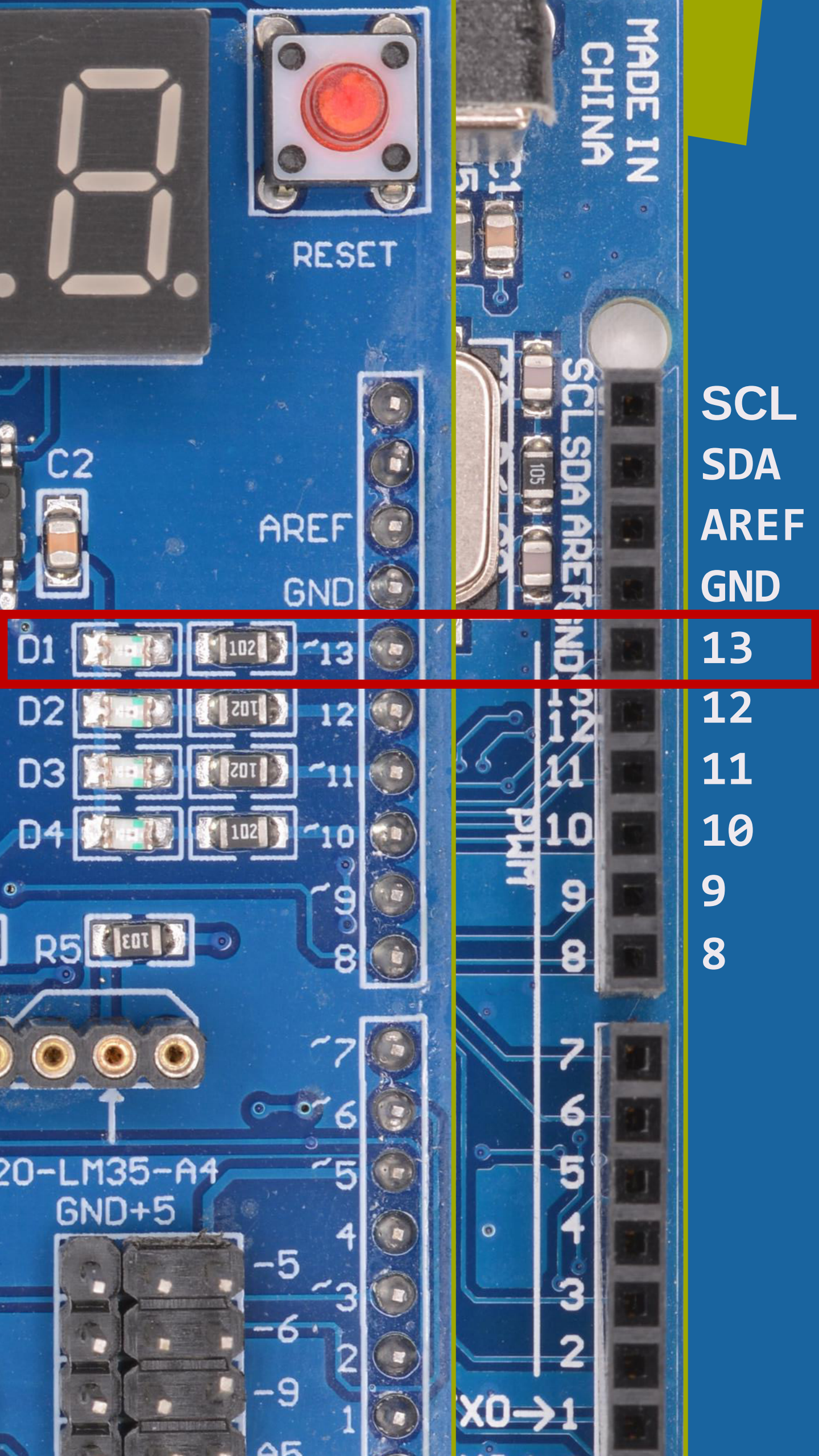




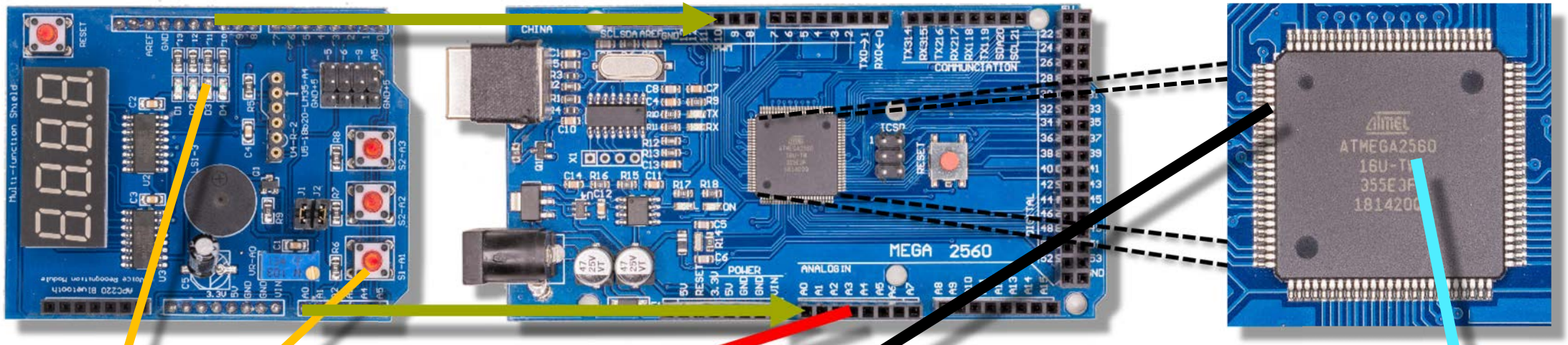


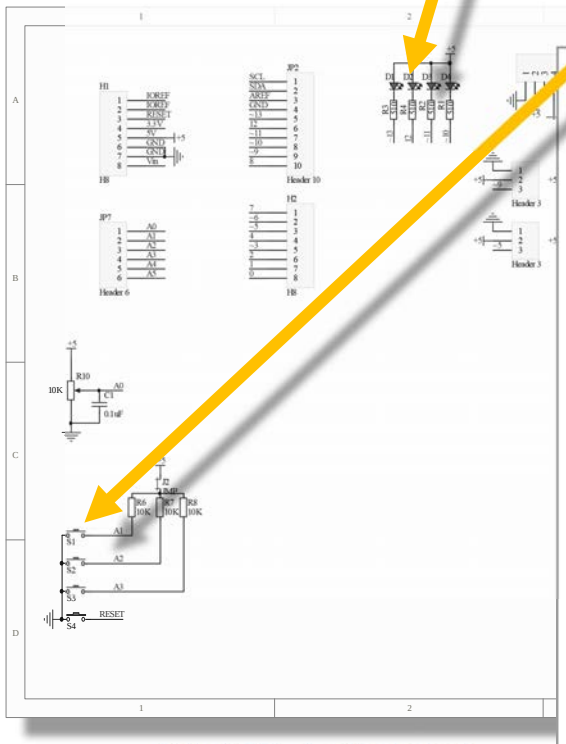
DE HAAGSE HOGESCHOOL

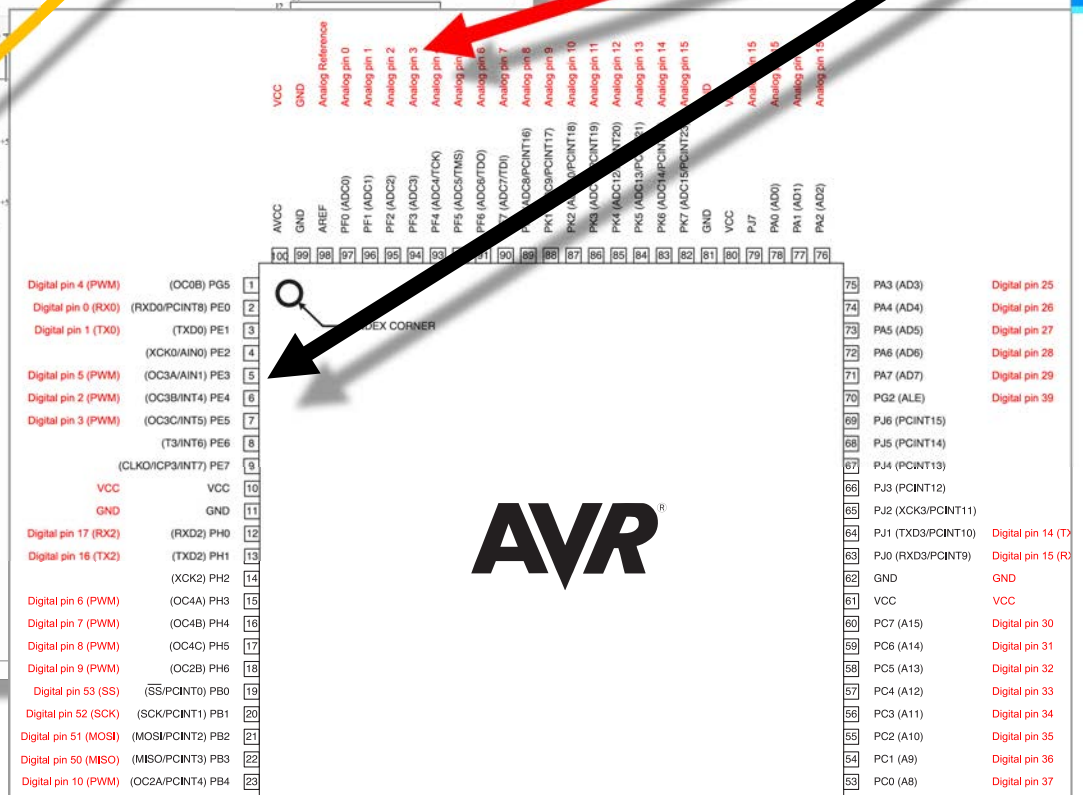

MADE IN
SCL SDA AREF GND
PWM
TX0→1
RX0←0
TX3 14
RX3 15
ICSP
RESET
ATMEGA2560
16U-TW
355E3F
181420Q
C1
R3
R2
R1
R4
C10
C8
C4
R10
R11
C7
R9
TX
RX
R12
R13
C13
X1
Q1
C14
R16
R15
C11
C12
R17
R18
L
ON
C5
DE HAAGSE
HOGESCHOOL


RESET
C2
AREF
GND
D1
D2
D3
D4
R5
~13
12
~11
~10
9
8
~7
6
~5
4
~3
2
1
MADE IN CHINA
SCL SDA AREF GND
PWM
20-LM35-A4
GND+5
~5
~6
~9
TX0→1
SCL
SDA
AREF
GND
13
12
11
10
9
8

shield “Arduino” microcontroller







Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V

8-bit Atmel Microcontroller with 16/32/64KB In-System Programmable Flash

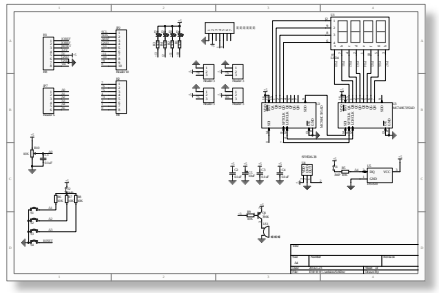
DATASHEET

Features

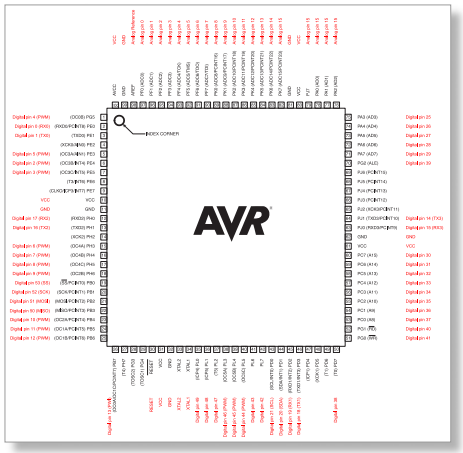
- High Performance, Low Power Atmel® AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
  - 135 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 16 MIPS Throughput at 16MHz
  - On-Chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
  - 64K/128K/256KBytes of In-System Self-Programmable Flash
  - 4Kbytes EEPROM
  - 8Kbytes Internal SRAM
  - Write/Erase Cycles:10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/ 100 years at 25°C
  - Optional Boot Code Section with Independent Lock Bits
    - In-System Programming by On-chip Boot Program
    - True Read-While-Write Operation
  - Programming Lock for Software Security
  - Endurance: Up to 64Kbytes Optional External Memory Space
- Atmel® QTouch® library support
  - Capacitive touch buttons, sliders and wheels
  - QTouch and QMatrix acquisition
  - Up to 64 sense channels
- JTAG (IEEE® std. 1149.1 compliant) Interface
  - Boundary-scan Capabilities According to the JTAG Standard
  - Extensive On-chip Debug Support
  - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - Four 16-bit Timer/Counter with Separate Prescaler, Compare- and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Four 8-bit PWM Channels
  - Six/Twelve PWM Channels with Programmable Resolution from 2 to 16 Bits (ATmega1281/2561, ATmega640/1280/2560)
  - Output Compare Modulator
  - 8/16-channel, 10-bit ADC (ATmega1281/2561, ATmega640/1280/2560)
  - Two/Four Programmable Serial USART (ATmega1281/2561, ATmega640/1280/2560)
  - Master/Slave SPI Serial Interface
  - Byte Oriented 2-wire Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
- I/O and Packages
  - 54/86 Programmable I/O Lines (ATmega1281/2561, ATmega640/1280/2560)
  - 64-pad QFN/MLF, 64-lead TQFP (ATmega1281/2561)

Week 1B - Leds change

Nummers en namen van pinnetjes

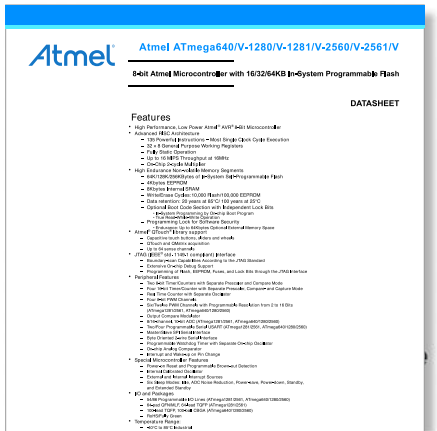


- Op de Arduino zijn alle pins genummerd.
- De pinnen zijn verbonden met de pinnen van de AVR



→ **Elektrisch schema shield en Arduino, pinout**

- Pinnetjes zijn gegroepeerd in banks van elk maximaal acht pinnen.
- Lezen en schrijven vanuit software gaat via registers PA, PB, PC, PD etc.


Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V

8-bit Atmel Microcontroller with 16/32/64KB In-System Programmable Flash

DATASHEET

Features

→ **AVR datasheet**

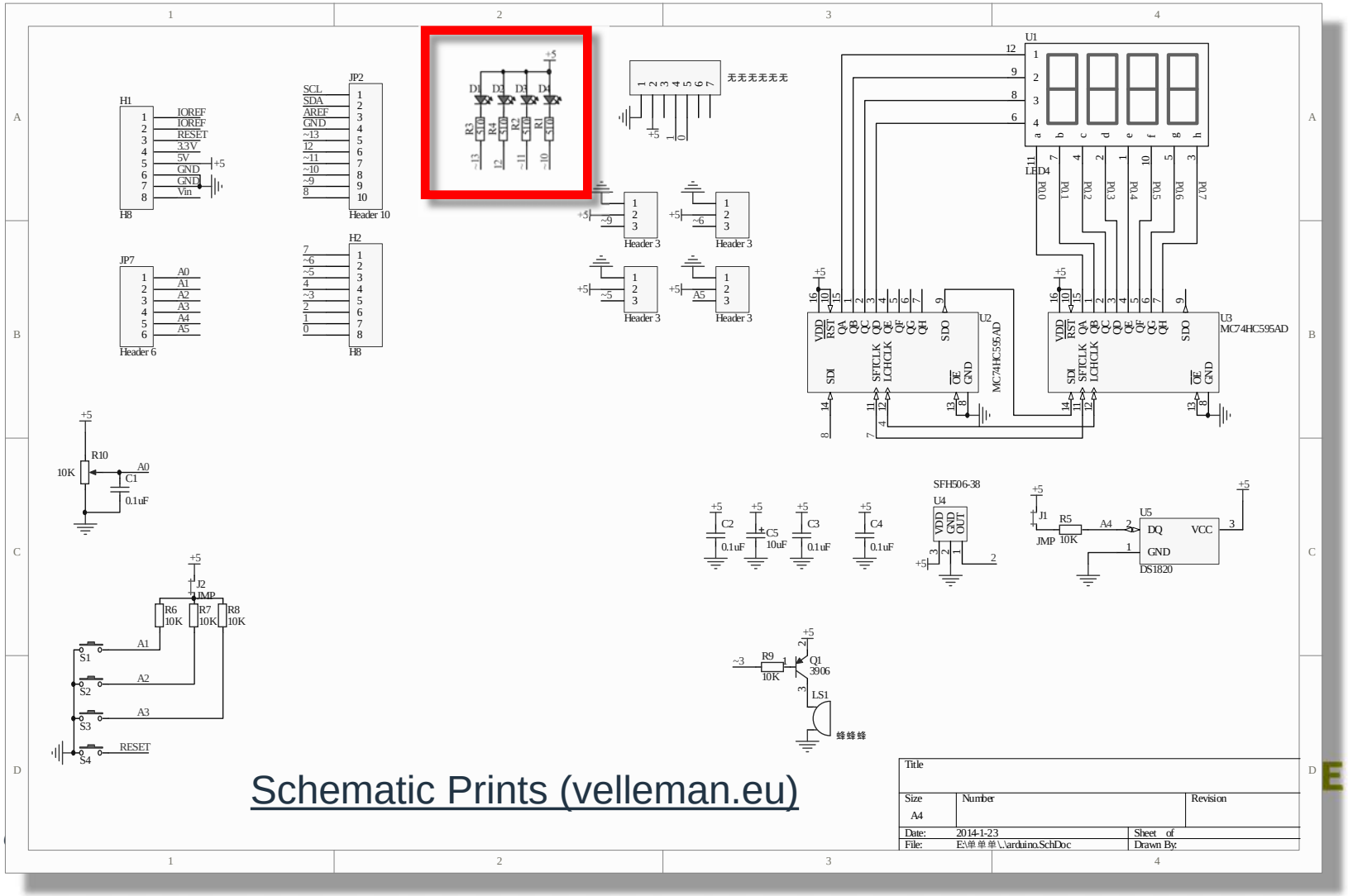
DE HAAGSE HOGESCHOOL

Configureer pin

We hebben een antwoord nodig op de volgende vragen:

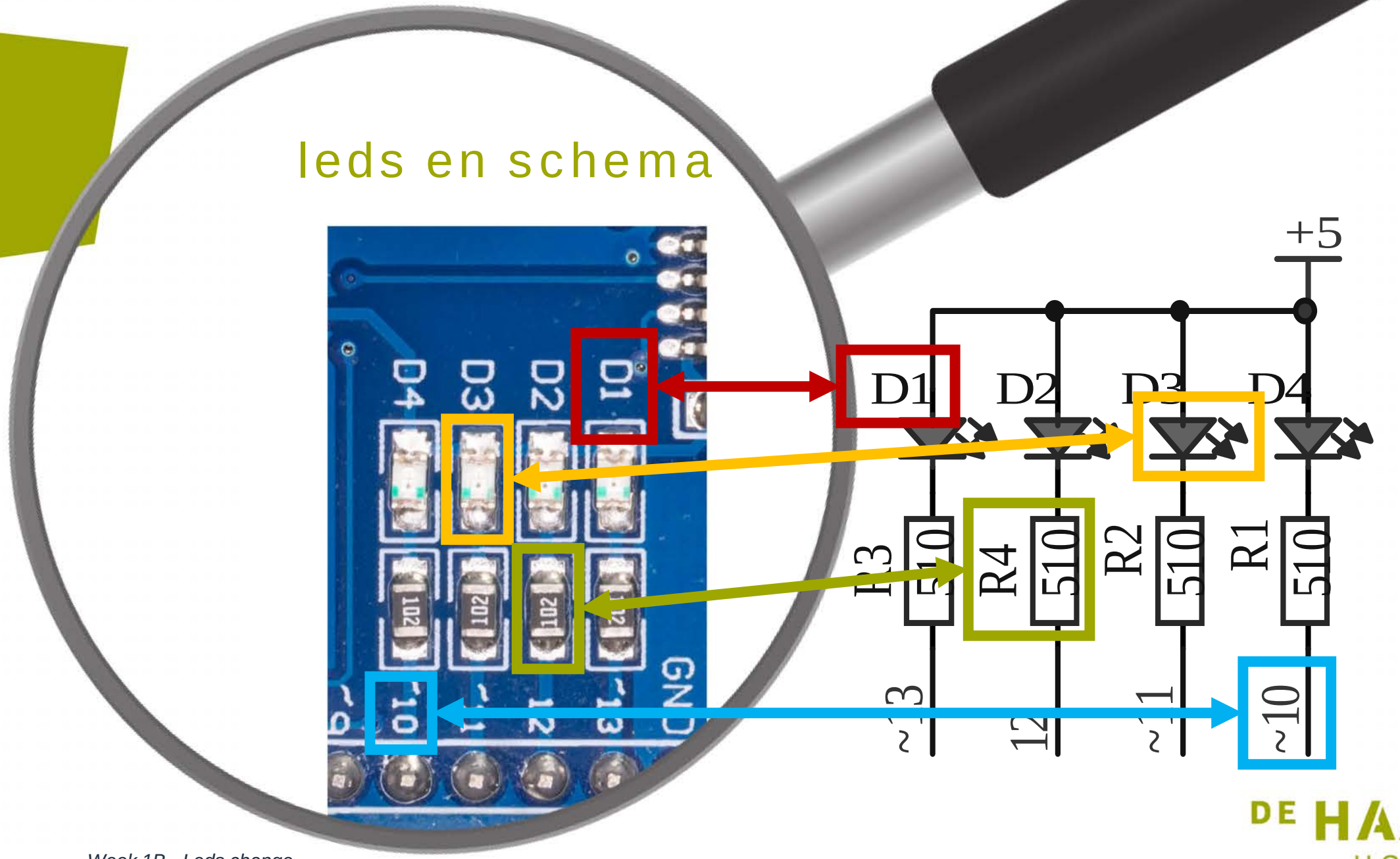
- Met welke pin is het ledje verbonden?
- Antwoord in elektrisch schema en pinout
- Met welke waarde van de pin gaat de led aan?
- Antwoord in elektrisch schema
- Hoe configureren we de pin als output pin?
- Antwoord in datasheet van de AVR

Elektrisch schema – leds



Schematic Prints (velleman.eu)

leds en schema

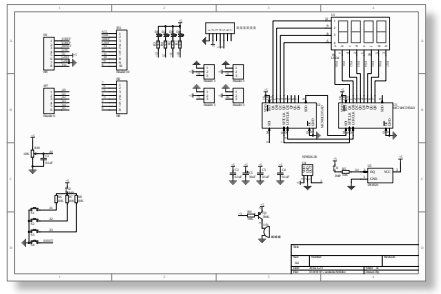


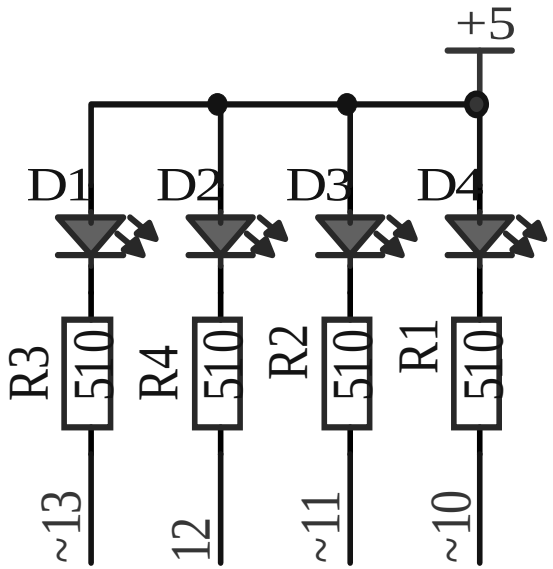
DE HAAGSE HOGESCHOOL

Configureer pin – pinnummer

Met welke AVR pin is ledje D1 verbonden?

- Elektrisch schema multi-IO shield → Arduino pin 13





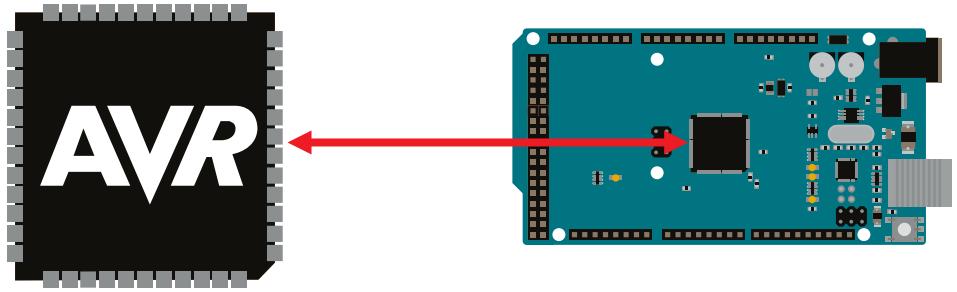
DE HAAGSE HOGESCHOOL

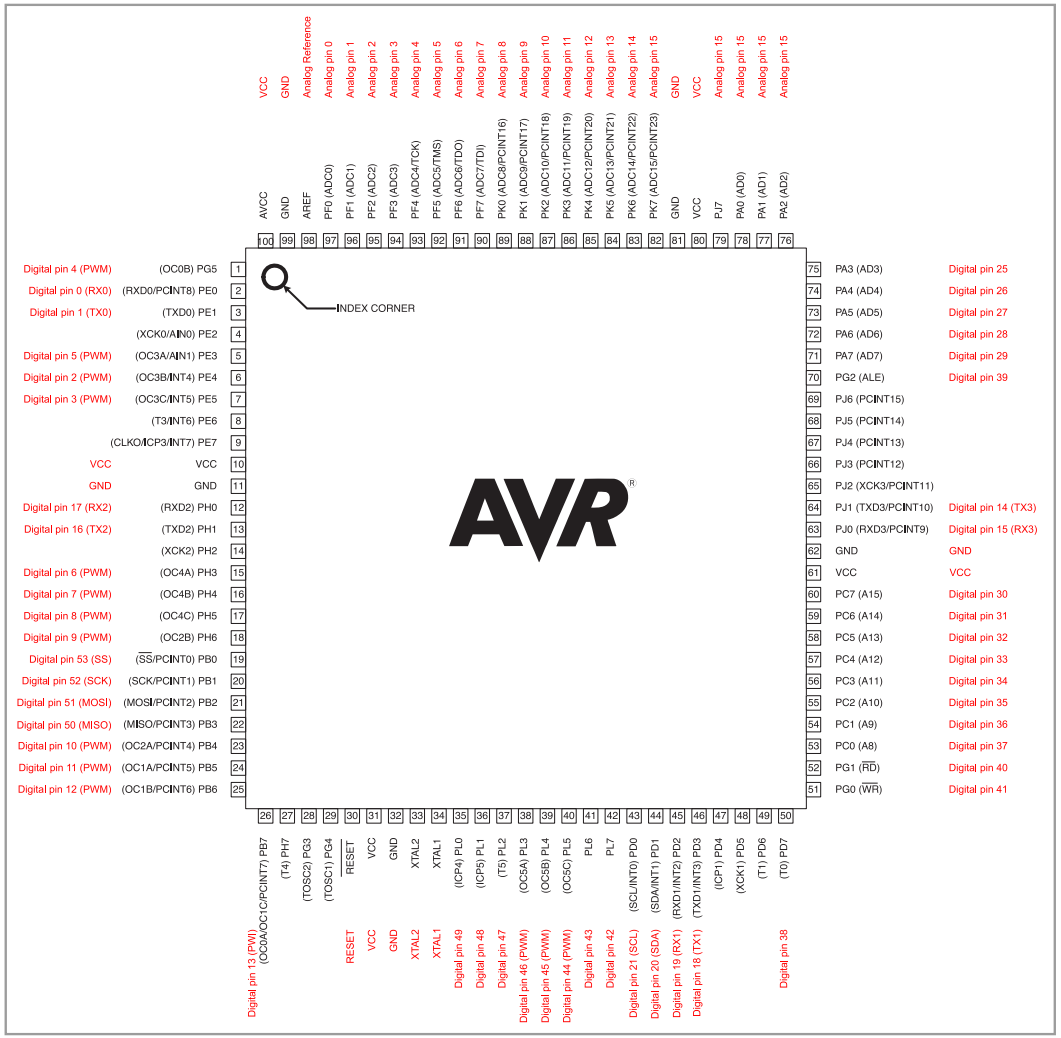
Pinout

- pinnen van de microcontroller?
- pinnen van “de Arduino”...

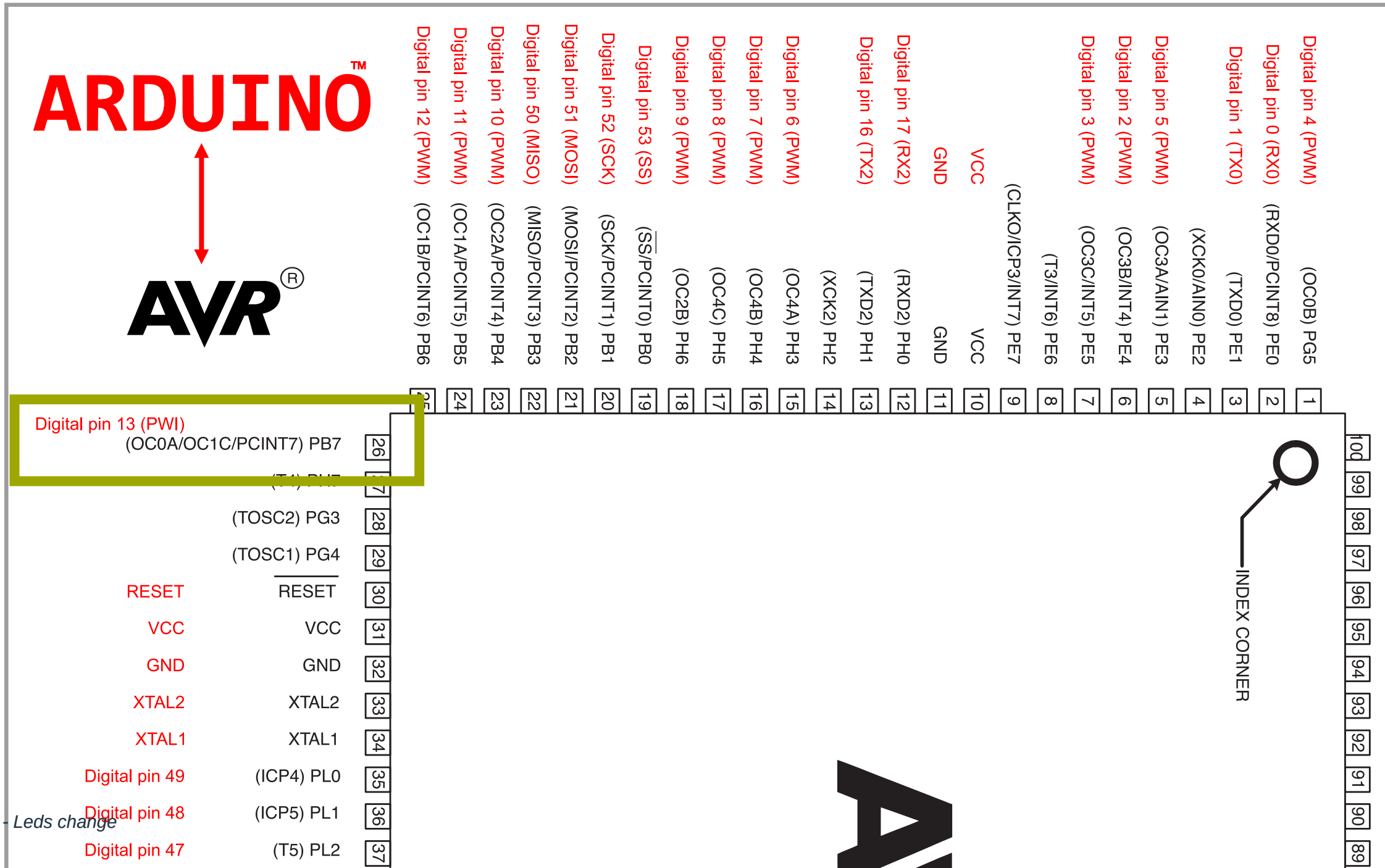
MICROCHIP

ARDUINO™





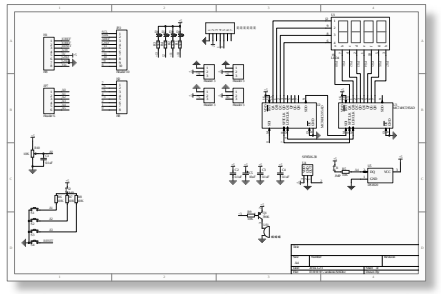
DE HAAGSE HOGESCHOOL

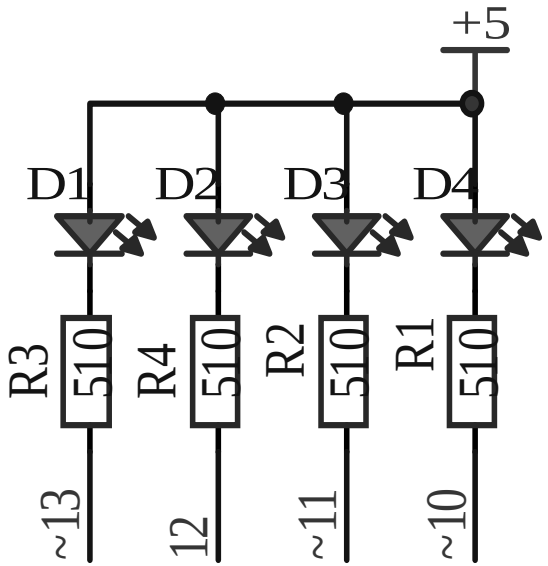

ARDUINO™
AVR®
Digital pin 13 (PWI)
(OC0A/OC1C/PCINT7) PB7
(TOSC2) PG3
(TOSC1) PG4
RESET
VCC
GND
XTAL2
XTAL1
Digital pin 49
Digital pin 48
Digital pin 47
RESET
VCC
GND
XTAL2
XTAL1
(ICP4) PL0
(ICP5) PL1
(T5) PL2
Digital pin 12 (PWM) (OC1B/PCINT6) PB6
Digital pin 11 (PWM) (OC1A/PCINT5) PB5
Digital pin 10 (PWM) (OC2A/PCINT4) PB4
Digital pin 50 (MISO) (MISO/PCINT3) PB3
Digital pin 51 (MOSI) (MOSI/PCINT2) PB2
Digital pin 52 (SCK) (SCK/PCINT1) PB1
Digital pin 53 (SS) (SS/PCINT0) PB0
Digital pin 9 (PWM) (OC2B) PH6
Digital pin 8 (PWM) (OC4C) PH5
Digital pin 7 (PWM) (OC4B) PH4
Digital pin 6 (PWM) (OC4A) PH3
VCC
GND
Digital pin 17 (RX2) (RXD2) PH0
Digital pin 16 (TX2) (TXD2) PH1
(XCK2) PH2
(CLKO/ICP3/INT7) PE7
Digital pin 3 (PWM) (OC3C/INT5) PE5
Digital pin 2 (PWM) (OC3B/INT4) PE4
Digital pin 5 (PWM) (OC3A/AIN1) PE3
(XCK0/AIN0) PE2
Digital pin 1 (TX0) (TXD0) PE1
Digital pin 0 (RX0) (RXD0/PCINT8) PE0
Digital pin 4 (PWM) (OC0B) PG5
INDEX CORNER

Configureer pin – aan/uit waarde

Gaat de led aan met ‘0’ of met ‘1’?

- Elektrisch schema multi-IO shield → ’0’ zet de led aan





DE HAAGSE HOGESCHOOL

13.2.1 Configuring the Pin

Each port pin consists of three register bits: DDxn, PORTxn, and PINxn. As shown in “Table 13-34 and Table 13-35 relates the alternate functions of Port L to the overriding signals shown in Figure 13-5 on page 73.” on page 95, the DDxn bits are accessed at the DDRx I/O address, the PORTxn bits at the PORTx I/O address, and the PINxn bits at the PINx I/O address.

The DDxn bit in the DDRx Register selects the direction of this pin. If DDxn is written logic one, Pxn is configured as an output pin. If DDxn is written logic zero, Pxn is configured as an input pin.

If PORTxn is written logic one when the pin is configured as an input pin, the pull-up resistor is activated. To switch the pull-up resistor off, PORTxn has to be written logic zero or the pin has to be configured as an output pin. The port pins are tri-stated when reset condition becomes active, even if no clocks are running.

If PORTxn is written logic one when the pin is configured as an output pin, the port pin is driven high (one). If PORTxn is written logic zero when the pin is configured as an output pin, the port pin is driven low (zero).

Atmel

Configureer pin – output pin

Datasheet hoofdstuk I/O-Ports:

- **DDRx** register (**Data Direction Register**) bepaalt input/output.
- Een **1** op de juiste plaats maakt de corresponderende pin **output**.
- **PORTx** register bepaalt de waarde op de output.
- Een **1** op de juiste plaats maakt de corresponderende pin **hoog**.

Bitnummering:

7	6	5	4	3	2	1	0

Configureer pin – output pin

Mega: PB7

DDRB = 0b10000000

PORTB = 0b10000000

Uno: PB5

DDRB = 0b00100000

PORTB = 0b00100000

Bitnummering:

7	6	5	4	3	2	1	0

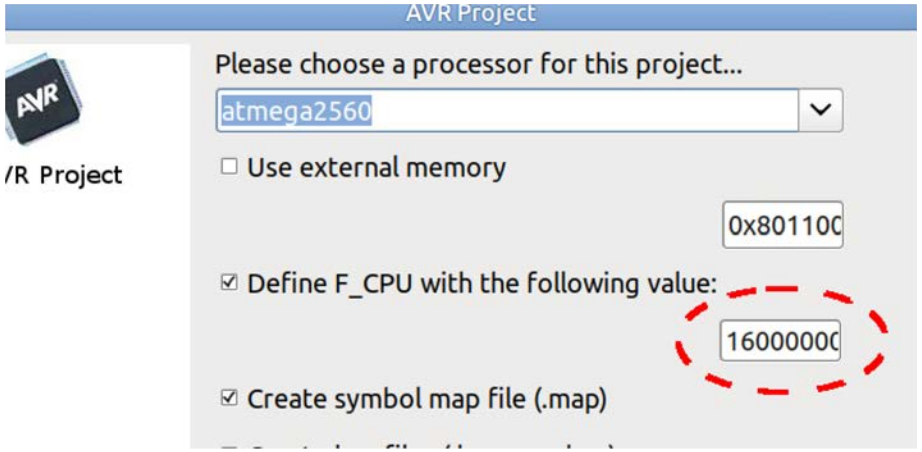
```
int main(void)
{
	// configureer pin
	DDRB = 0b10000000;		// voor de Mega
	while (1)
	{
		// Zet led aan
		PORTB = 0b00000000;	// aanzetten met '0'
		// Wacht 500 ms
		// Zet led uit
		PORTB = 0b10000000;	// uitzetten met '1'
		// Wacht 500 ms
	}
}
```

500 ms wachten

Zet bovenaan `#include <util/delay.h>`

Gebruik de functie **`_delay_ms (<aantal ms>)`**

Deze functie vereist de definitie van **`F_CPU`**



DE HAAGSE HOGESCHOOL

Include files nodig

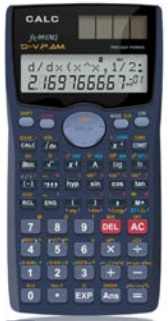
```
#include <avr/io.h>          // voor gebruik DDRx
#include <util/delay.h>      // voor delay_ms()
```

```c
int main(void)
{
    // configureer pin
    DDRB = 0b10000000;          // voor de Mega
    while (1)
    {
        // Zet led aan
        PORTB = 0b00000000;   // aanzetten met '0'
        _delay_ms(500);
        // Zet led uit
        PORTB = 0b10000000;   // uitzetten met '1'
        _delay_ms(500);
    }
}
```

Top 3 belangrijke dia's

int
printf en scanf
%d en &
+ - x / %
=

```c
// Invoer
printf("Hoeveel studenten zijn er?\n");
scanf("%d", &aantalDelers);
aantalDelers = aantalDelers + 1;

// Verwerking
int aantalSnoep = aantalZakken * snoepInZak;
int snoepPerPersoon = aantalSnoep / aantalDelers - snoepVoorMa;

// Uitvoer
printf("Snoepjes per persoon: %d\n", snoepPerPersoon);
printf("Snoepjes over: %d\n", aantalSnoep % aantalDelers);
```



shield, "Arduino" en AVR
elektrisch schema
pinout
data sheet

```c
int main(void)
{
    // configureer pin
    DDRB = 0b10000000;          // voor de Mega
    while (1)
    {
        // Zet led aan
        PORTB = 0b00000000;   // aanzetten met '0'
        _delay_ms(500);
        // Zet led uit
        PORTB = 0b10000000;   // uitzetten met '1'
        _delay_ms(500);
```

DDR en PORT
bitjes 0 of 1

DE HAAGSE HOGESCHOOL

Samenvatting

- Beslissen met `if (...) { ... }`
- Oneindig lang herhalen met `while (1) { ... }`
- AVR pin instellen als output met DDRx
- Uitgangswaarde van AVR pin instellen met PORTx
- Gebruik elektrisch schema en Arduino pinout om pinnummers te bepalen

let's change

YOU. US. THE WORLD.

1010 0100.