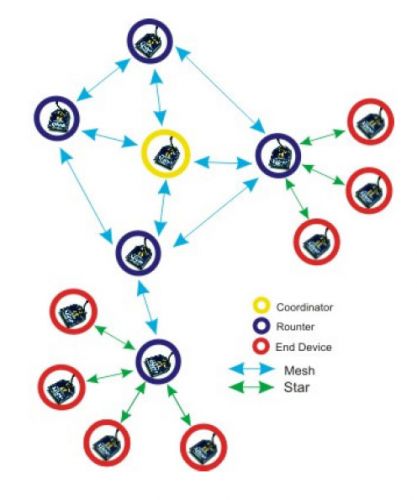


Onderzoek sensornetwerken

Internet of Things



19 september 2016

Eddy Luursema

Inhoudsopgave

[1 Netwerken 4](#_Toc462062685)

[1.1 Inleiding 4](#_Toc462062686)

[1.2 Uitvoering 2015-2016 P1 4](#_Toc462062687)

[1.3 Klacht studenten 4](#_Toc462062688)

[1.4 Uitvoering 2016-2017 P1 4](#_Toc462062689)

[2 Onderzoek sensornetwerken 5](#_Toc462062690)

[2.1 Onderzoeksvragen 5](#_Toc462062691)

[2.2 Inleveren 5](#_Toc462062692)

[3 Onderzoek snelheid en delay NRF24 6](#_Toc462062693)

[3.1 FTDI 6](#_Toc462062694)

[3.2 Chinese Arduino-klonen 6](#_Toc462062695)

[3.3 Windows 10 7](#_Toc462062696)

[3.4 Gebruik van slechts één laptop 7](#_Toc462062697)

[3.5 Opstelling 8](#_Toc462062698)

[3.6 Installeren NRF24 library 8](#_Toc462062699)

[3.7 Transmit – en receive programma 9](#_Toc462062700)

[3.8 Nogmaals op 1 laptop 12](#_Toc462062701)

[3.9 Onderzoeksdeelvragen 12](#_Toc462062702)

[4 Onderzoek netwerkarchitectuur 13](#_Toc462062703)

[4.1 Meetopstelling 13](#_Toc462062704)

[4.2 Gebruik Pi 14](#_Toc462062705)

[4.3 Software mesh netwerk 14](#_Toc462062706)

[4.4 Wissen EEPROM 14](#_Toc462062707)

[4.5 Aansluiten gateway 15](#_Toc462062708)

[4.6 Installeren MySensors library 15](#_Toc462062709)

[4.7 Gateway ‘programmeren’ 16](#_Toc462062710)

[4.8 Installeren end point 17](#_Toc462062711)

[4.9 MySensors seriële protocol 17](#_Toc462062712)

[4.10 Bestuderen debug-informatie 17](#_Toc462062713)

[4.11 Sniffen netwerkverkeer 17](#_Toc462062714)

[4.12 Installeren routers 18](#_Toc462062715)

[4.13 Onderzoeksvragen 18](#_Toc462062716)

[4.14 Facultatieve opdracht 19](#_Toc462062717)

[4.15 Facultatieve opdrachten: Ethernet gateway 19](#_Toc462062718)

[4.16 Facultatieve opdracht 19](#_Toc462062719)

[4.17 Faculatieve opdracht 19](#_Toc462062720)

[4.18 Facultatieve opdracht: Xbee 19](#_Toc462062721)

# Netwerken

## Inleiding

In het de course IoT-I (internet of Things –Internet) course worden onderzoeksvaardigheden aangeleerd en geoefend. Binnen IoT-I wordt vooral aandacht besteed aan de component ‘Lab’ uit het Triangulatie-framework (<http://www.nioc2013.nl/wp-content/uploads/2013/12/nioc-paper-definitief-KvT.pdf>).

## Uitvoering 2015-2016 P1

De studenten kregen vanuit de course IoT-I een hoofd-onderzoeksvraag aangereikt. Zij moesten daarop via een literatuurstudie de hoofdvraag proberen te beantwoorden via een aantal deeloplossingen. Deze deeloplossingen werden onderling vergeleken via een zelf samen te stellen set criteria. Nadat een overzicht van alle deeloplossingen en criteria waren opgesteld werd van een aantal deeloplossingen via literatuuronderzoek een shortlist worden gemaakt van geschikte deeloplossingen voor het laboratoriumonderzoek. In feite werden de componenten ‘Bieb’ en ‘Lab’ van het triangulatie-framework uitgevoerd.

## Klacht studenten

Omdat in het onderzoek van de course IoT-T ook al een hoofdvraag middels literatuurstudie en een shortlist werd gedaan, ervaarden de studenten dit als dubbel. Ook werden beide onderzoeken uitgevoerd in de laatste weken van de course wat tot nadelige inspanningsresultaten leidde voor het ene of het andere onderzoek; meestal was het onderzoek IoT-I het slachtoffer.

## Uitvoering 2016-2017 P1

Besloten is om in deze uitvoering van het onderzoek binnen IoT-I alleen uit te voeren.

Het aansluiten/programmeren van de gebruikte XBee’s kostte erg veel tijd in verhouding tot het feitelijke onderzoek naar doorvoersnelheid en meshing. Bovendien moest er in 2015-P4 een 2e laboratorium voor IoT worden ingericht en bij de inventarisatie bleek, dat de XBee’s intussen rond de €29 per stuk kosten. Vanwege de millimeter-behuizing heb je een aanpassingsprint van €10 of een USB-kaartje van €30 nodig. Voor onderzoek naar het zelfherstellend gedrag zijn tenminste 3 Xbee’s + aansluitprint nodig en dat heeft het ontwikkelteam doen besluiten de XBee te laten vervallen.

# Onderzoek sensornetwerken

## Onderzoeksvragen

Welke mogelijkheden bestaan er om een netwerk aan te leggen tussen rijdende kleine robots, gebaseerd op Arduino-technologie, waarmee deze diverse sensorwaarden waaronder hun snelheid en positie kunnen doorgeven aan een centrale server?

Inventariseer een aantal criteria en oplossingsrichtingen voor deze onderzoekshoofdvraag.

Deze gaan we klassikaal opstellen/bespreken.

## Inleveren

**Documentatie**

Bij het laboratoriumonderzoek zal voldoende documentatie moeten worden gemaakt zodat de proeven gereproduceerd kunnen worden. Verder dient de software te voldoen aan de standaarden zoals aangeleerd bij de course ESD uit de propedeuse. De gemaakte ontwerpen, schema’s en software moeten ingecheckt worden in SVN. Additionele software (dus vooral niet-standaard libraries en niet de Arduino-IDE, Visual Studio, Processing en Linux) moeten aan de repository worden toegevoegd met installatiehandleiding.

Het eindverslag bevat de onderzoeksvragen, een gedeeltelijke uitwerking, gemaakte ontwerpen, schema’s en software, een opsomming van meetresultaten en een conclusie.

# Onderzoek snelheid en delay NRF24

Onderstaande is deels overgenomen uit <https://arduino-info.wikispaces.com/Nrf24L01-2.4GHz-ExampleSketches#bm1>

## FTDI

Om een Arduino Pro mini te kunnen ‘programmeren’ heb je een speciale interface nodig: de FTDI Basic Breakout adapter. Je kunt ook een andere Arduino Uno(met grote chip) gebruiken.



**PAS OP: er zijn Arduino mini’s van 3.3V en 5V. De FTDI bevat een schakelaar voor het instellen van de juiste spanning !!**

**PAS OP: vanwege de prijs en de frequentie van het gebruik van de adapter zijn er in de centrale laboratoriumdoos maar enkele exemplaren van de FTDI Basic Breakout adapter aanwezig.**

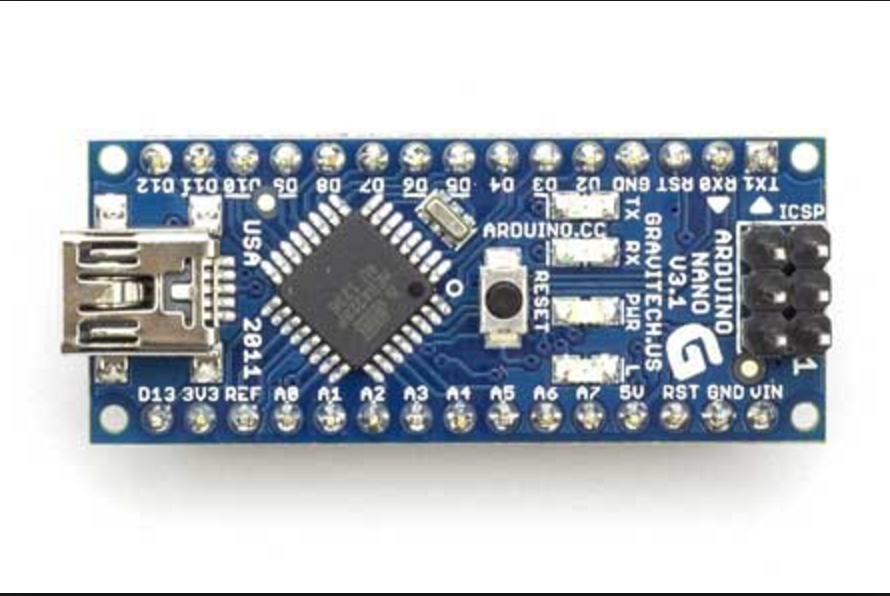
Bronnen:

<https://www.sparkfun.com/products/9873>

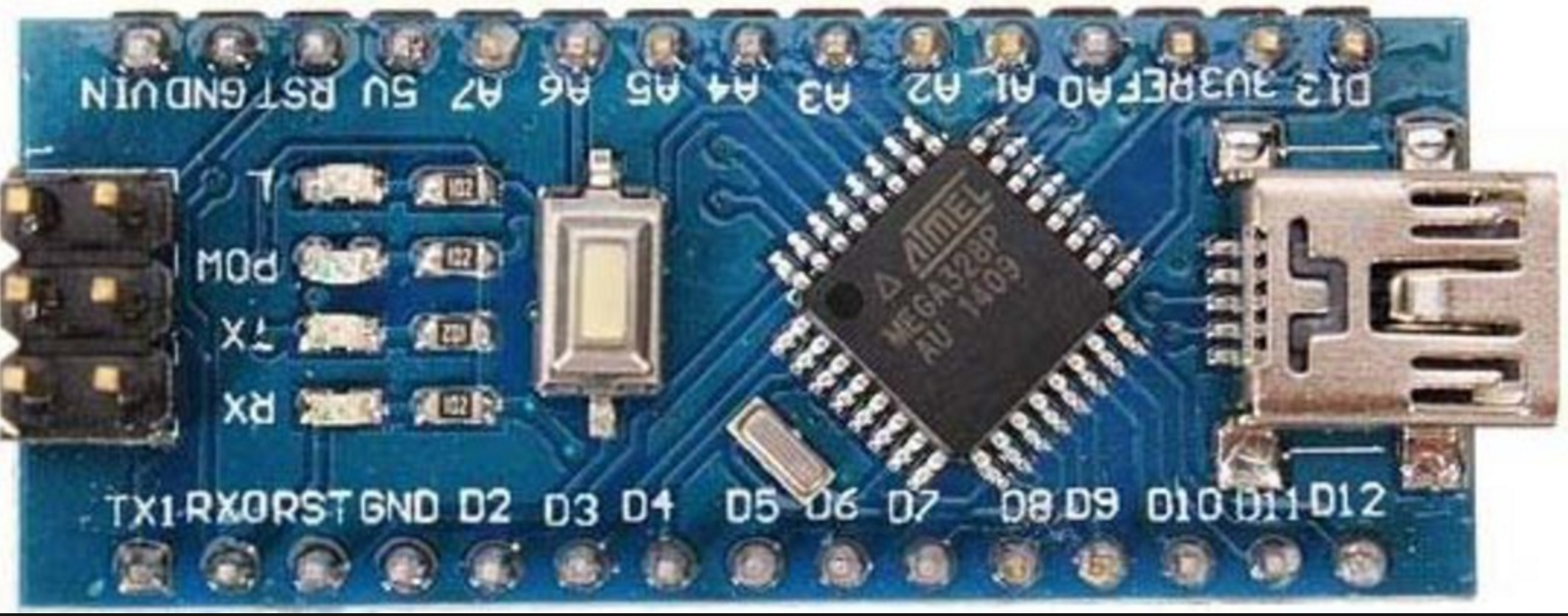
<https://www.arduino.cc/en/Guide/ArduinoMini>

## Chinese Arduino-klonen

Om geld te besparen hebben we via een Chinese site klonen van de Arduino Nano gekocht. Deze zijn te herkennen aan het ontbreken van de tekst ‘Arduino Nano V3.1’:



Figuur 1Echte

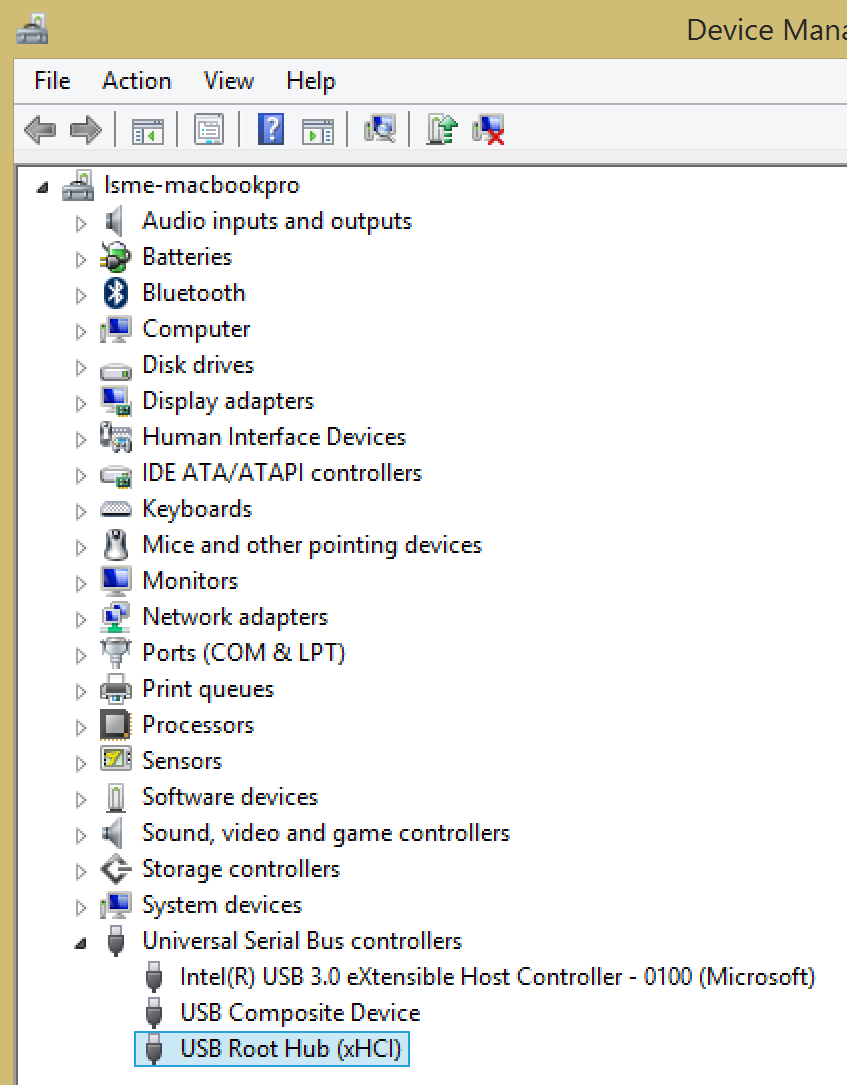


Figuur 2 Kloon

Bij het gebruik van bepaalde Arduino’s komt het voor, dat het een Chinese kloon betreft met een andere seriële chip (zowel de Nano als de FTDI Basic Breakout adapter). Hiervoor moet op de laptop een nieuwe driver worden geïnstalleerd. Zie hiervoor:

<http://kig.re/2014/12/31/how-to-use-arduino-nano-mini-pro-with-CH340G-on-mac-osx-yosemite.html>

Tip: wij ondervonden het probleem, dat het menu-item ‘Other devices’ niet verscheen:



Gewoon installeren via setup.exe vanuit het zip-bestand dat we op de desktop hadden geplaatst.

## Windows 10

Het lijkt er op dat na het installeren van de CH340-driver de oorspronkelijke driver voor de FT232-chip is verwijderd. Dit valt te bekijken onder Device manager – Ports. Dit houdt in dat je ‘echte’ Arduino Nano’s niet meer kunt programmeren.

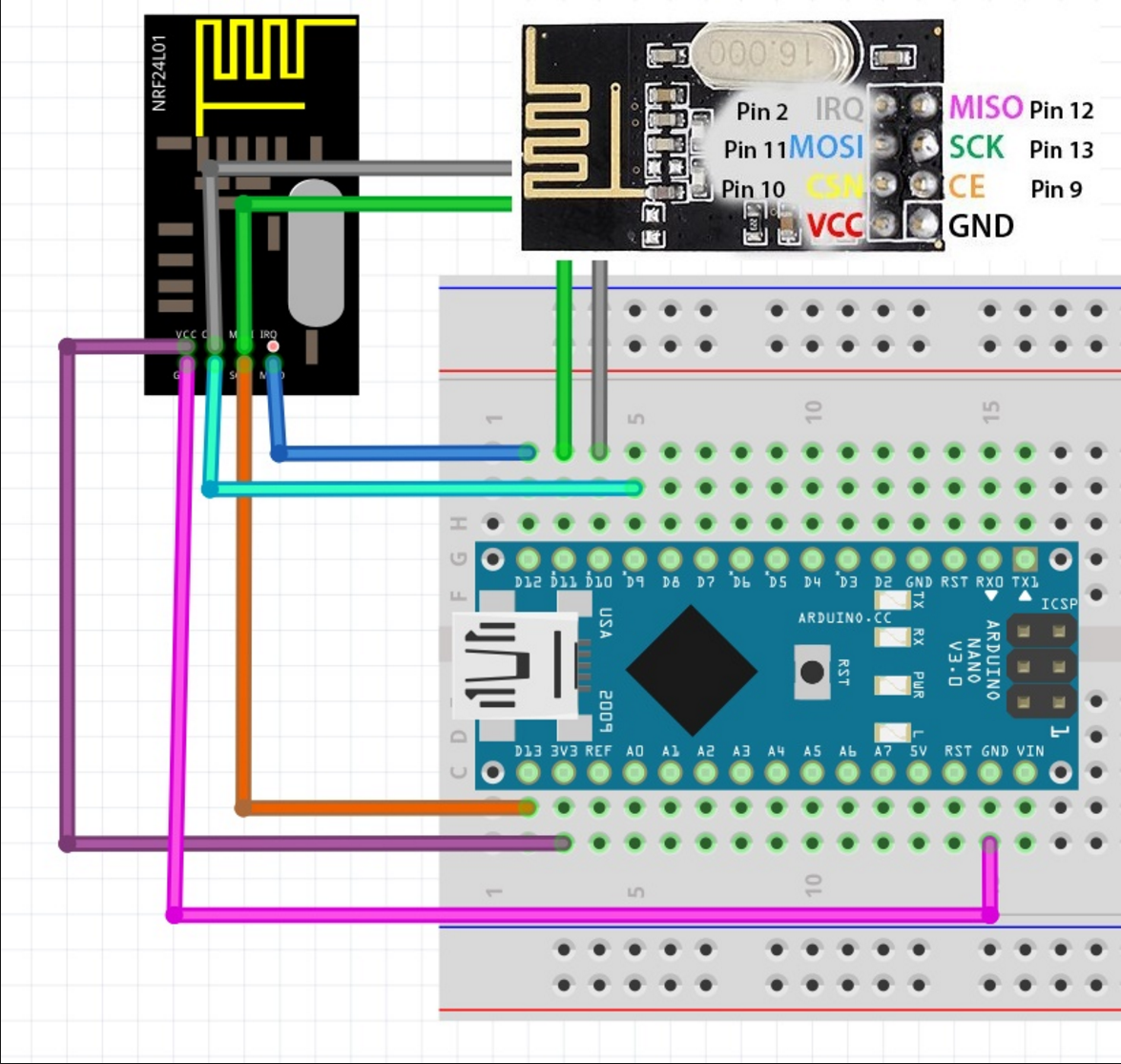
Ga eventueel naar [www.ftdichip.com/Drivers/VCP.htm](http://www.ftdichip.com/Drivers/VCP.htm) en kies de Windows-variant om deze alsnog weer toe te voegen.

## Gebruik van slechts één laptop

Bij gebruik van slechts één laptop voor gateway en endpoint is het probleem, dat de Arduino-ontwikkelomgeving lastig kan omspringen met twee USB-poorten. Op een Windows 8 machine is het ons wel gelukt. Tip: plaats de linker en rechter IDE overeenkomstig de linker en rechter Arduino anders word je gek.

Onder OS-X is dit standaard niet mogelijk; OS-X laat geen 2 instanties toe van de Arduino-IDE. Er is een lelijke oplossing: kopieer de Arduino.app in je map Programma’s en start beide instanties op.

## Opstelling

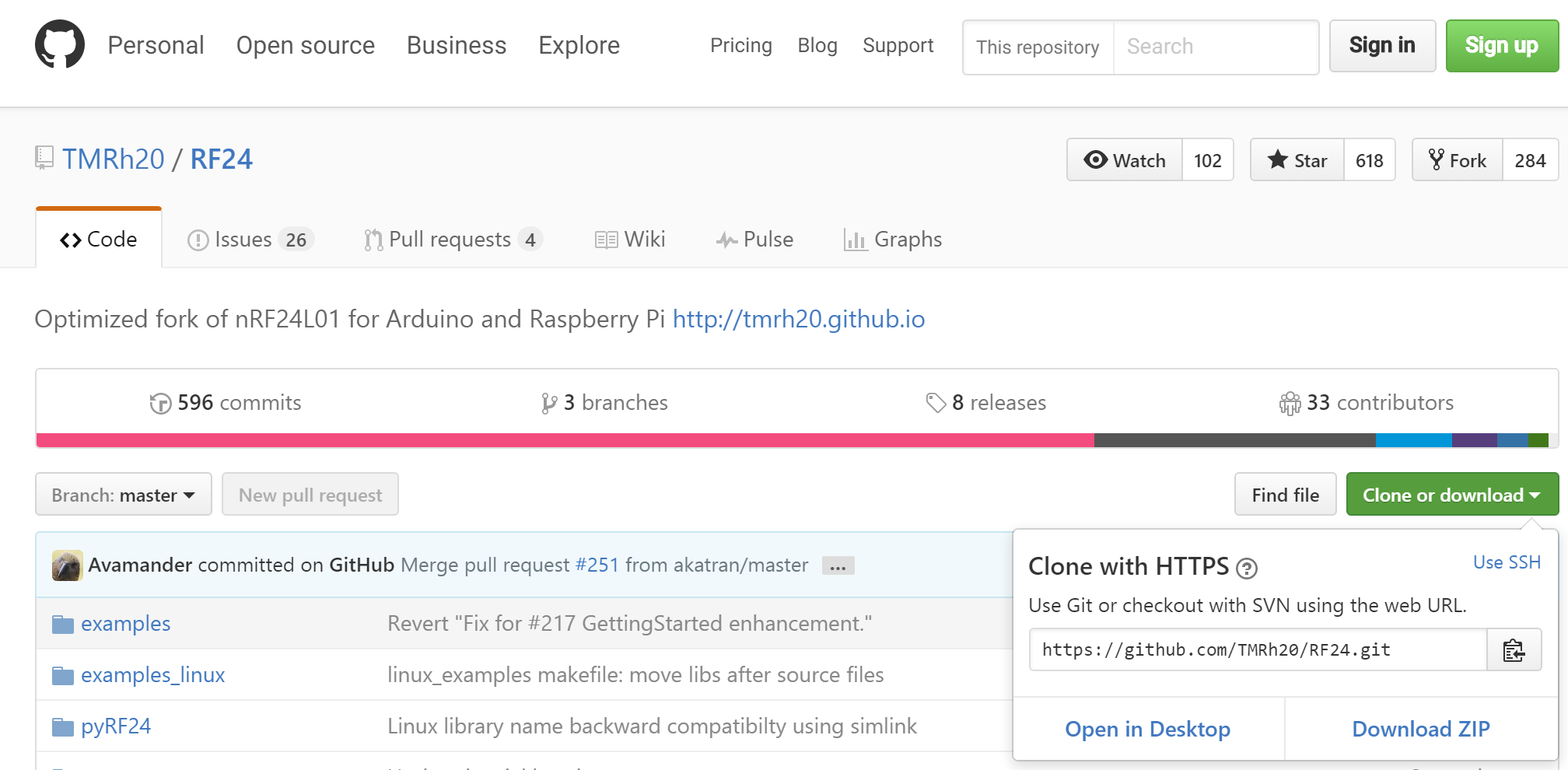


**Tip: laat de 8 kabeltjes voor een NRF aan elkaar ‘geplakt’ zitten. Dit bevordert het overzicht.**

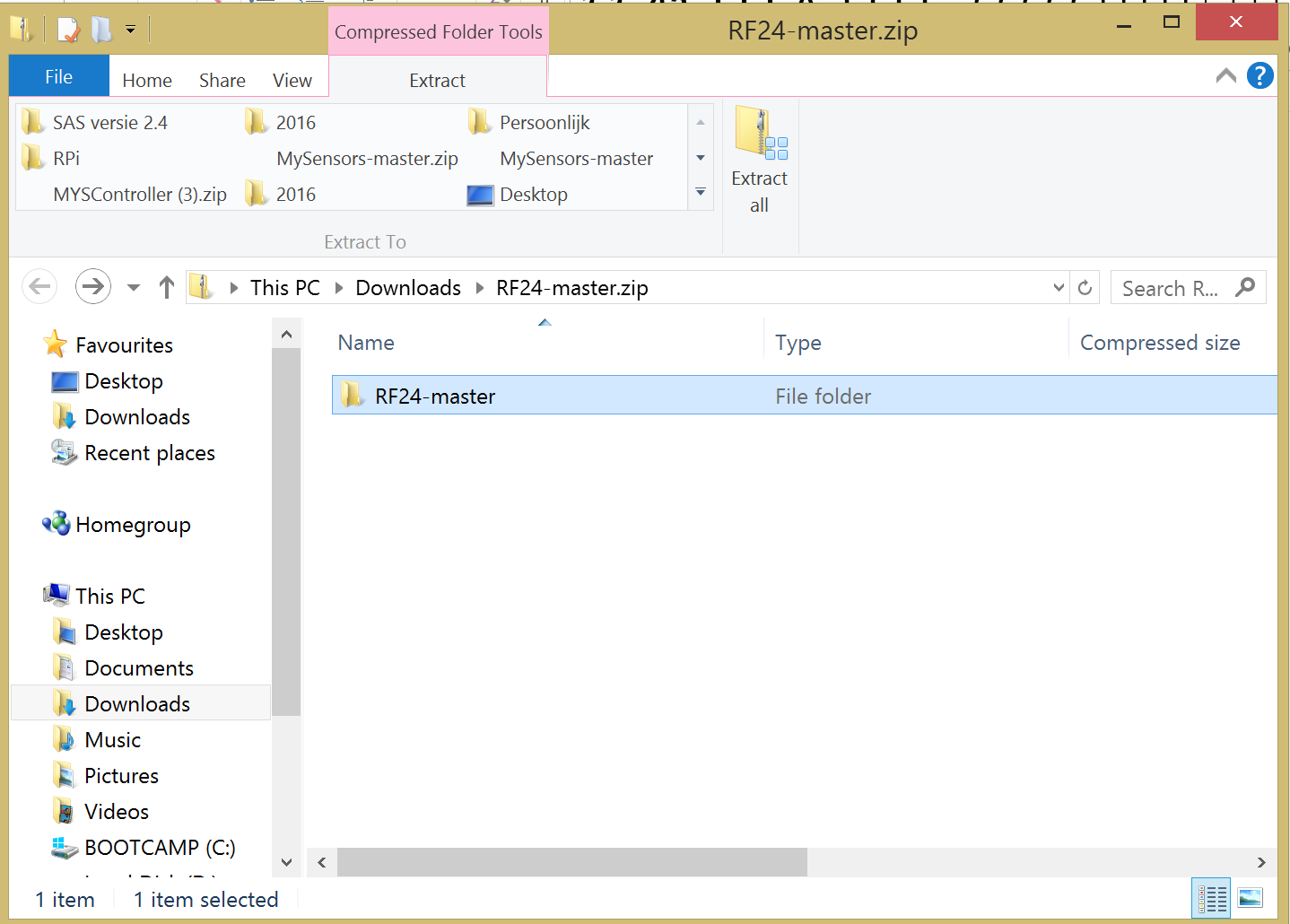
## Installeren NRF24 library

Ga naar <https://github.com/TMRh20/RF24>

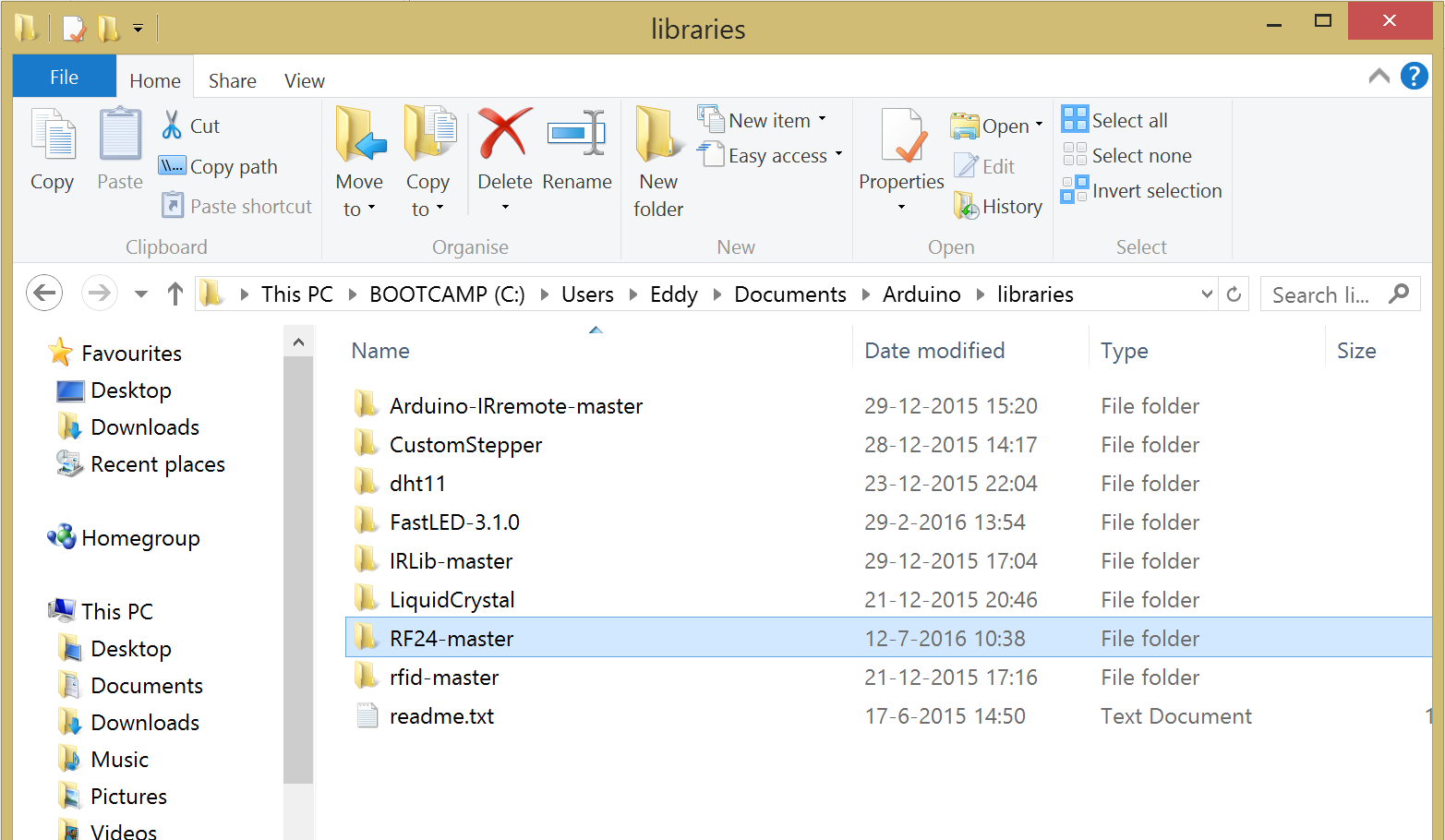
Download het ZIP-bestand



Open ZIP-bestand:



En kopieer de map ‘RF24-master’ naar je persoonlijke library-map onder Windows:

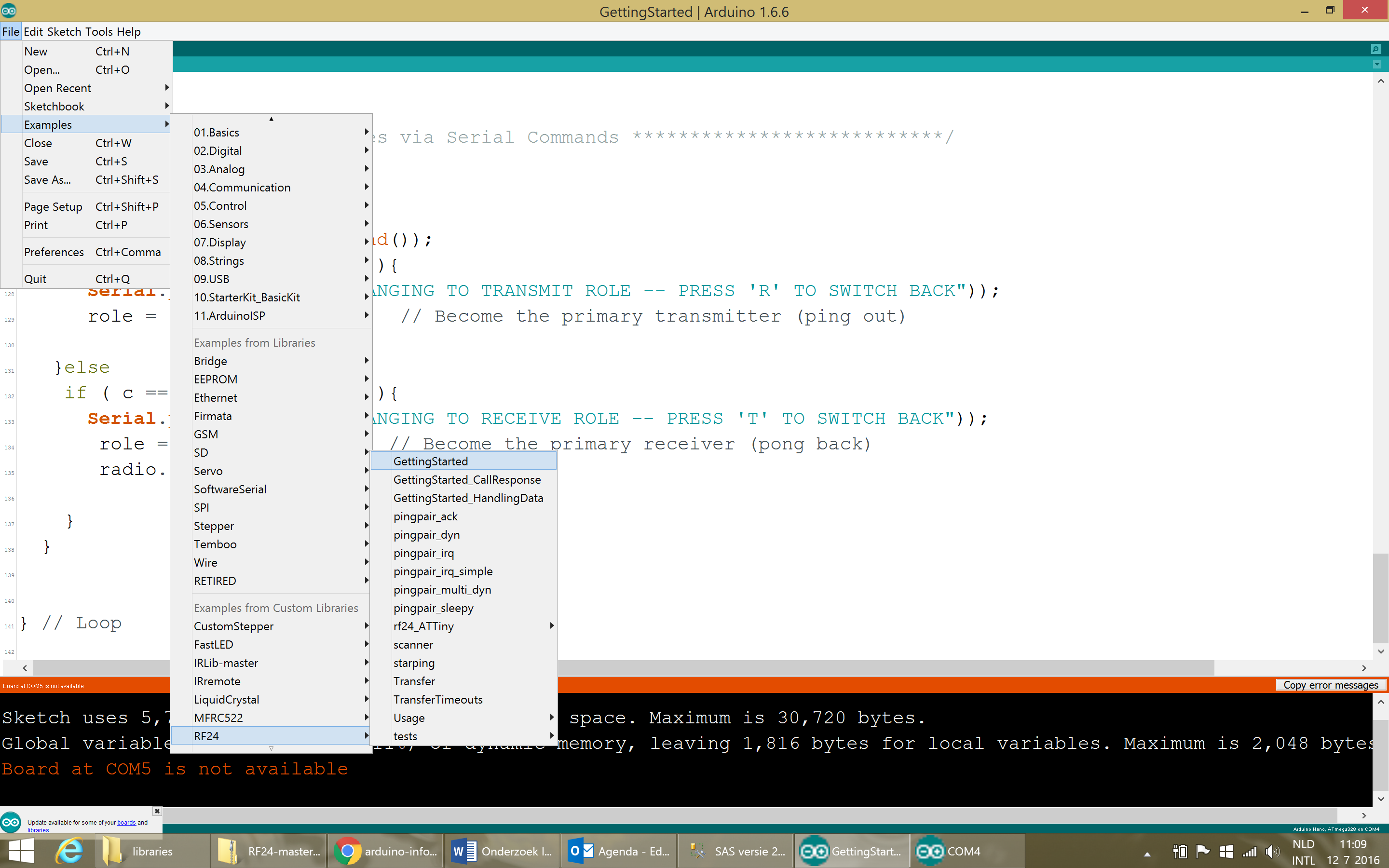


Sluit eventueel je Arduino-IDE en open opnieuw.

## Transmit – en receive programma

Laad uit de ‘Examples’ van de NRF24 -library het voorbeeld ‘GettingStarted’:

Stel het kanaalnummer in: 71 t/m 82 voor de practicumgroepen 1 t/m 12 m.b.v. radio.setChannel()



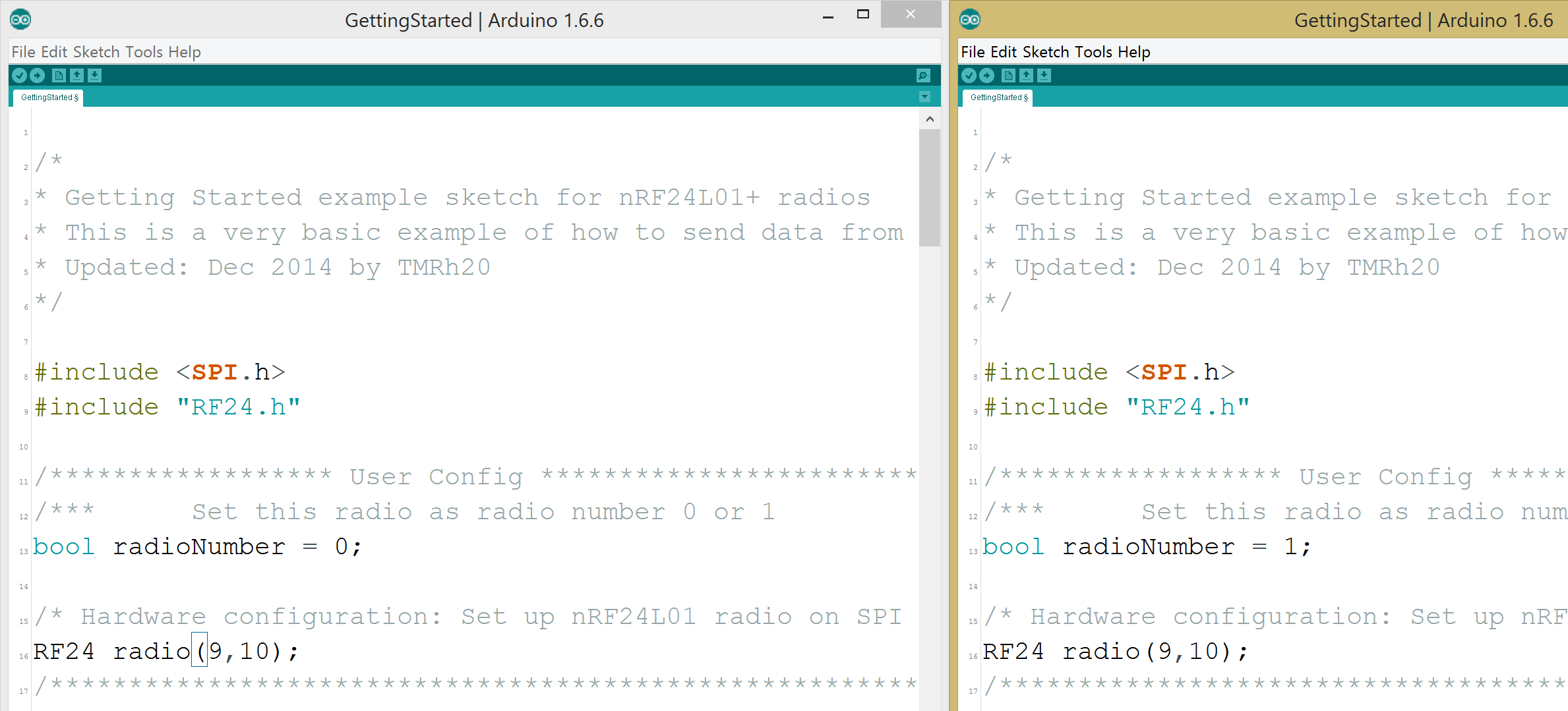
Zorg dat je op de ene kant radioNumber = 0 en de andere kant radioNumber = 1 instelt.

Let op: in bovenstaande hardware-aansluiting zitten de CE en CSN op pin 9 en 10 dus moet je de radio-module initialiseren met RF24 radio(9,10):

|  |
| --- |
| /\*  \* Getting Started example sketch for nRF24L01+ radios  \* This is a very basic example of how to send data from one node to another  \* Updated: Dec 2014 by TMRh20  \*/  #include <SPI.h>  #include "RF24.h"  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* User Config \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  /\*\*\* Set this radio as radio number 0 or 1 \*\*\*/  bool radioNumber = 1;  /\* Hardware configuration: Set up nRF24L01 radio on SPI bus plus pins 7 & 8 \*/  RF24 radio(9,10);  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  byte addresses[][6] = {"1Node","2Node"};  // Used to control whether this node is sending or receiving  bool role = 0;  void setup() {  Serial.begin(115200);  Serial.println(F("RF24/examples/GettingStarted"));  Serial.println(F("\*\*\* PRESS 'T' to begin transmitting to the other node"));    radio.begin();  // Set the PA Level low to prevent power supply related issues since this is a  // getting\_started sketch, and the likelihood of close proximity of the devices. RF24\_PA\_MAX is default.  radio.setPALevel(RF24\_PA\_LOW);    // Open a writing and reading pipe on each radio, with opposite addresses  if(radioNumber){  radio.openWritingPipe(addresses[1]);  radio.openReadingPipe(1,addresses[0]);  }else{  radio.openWritingPipe(addresses[0]);  radio.openReadingPipe(1,addresses[1]);  }    // Start the radio listening for data  radio.startListening();  }  void loop() {      /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Ping Out Role \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  if (role == 1) {    radio.stopListening(); // First, stop listening so we can talk.      Serial.println(F("Now sending"));  unsigned long start\_time = micros(); // Take the time, and send it. This will block until complete  if (!radio.write( &start\_time, sizeof(unsigned long) )){  Serial.println(F("failed"));  }    radio.startListening(); // Now, continue listening    unsigned long started\_waiting\_at = micros(); // Set up a timeout period, get the current microseconds  boolean timeout = false; // Set up a variable to indicate if a response was received or not    while ( ! radio.available() ){ // While nothing is received  if (micros() - started\_waiting\_at > 200000 ){ // If waited longer than 200ms, indicate timeout and exit while loop  timeout = true;  break;  }  }    if ( timeout ){ // Describe the results  Serial.println(F("Failed, response timed out."));  }else{  unsigned long got\_time; // Grab the response, compare, and send to debugging spew  radio.read( &got\_time, sizeof(unsigned long) );  unsigned long end\_time = micros();    // Spew it  Serial.print(F("Sent "));  Serial.print(start\_time);  Serial.print(F(", Got response "));  Serial.print(got\_time);  Serial.print(F(", Round-trip delay "));  Serial.print(end\_time-start\_time);  Serial.println(F(" microseconds"));  }  // Try again 1s later  delay(1000);  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Pong Back Role \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  if ( role == 0 )  {  unsigned long got\_time;    if( radio.available()){  // Variable for the received timestamp  while (radio.available()) { // While there is data ready  radio.read( &got\_time, sizeof(unsigned long) ); // Get the payload  }    radio.stopListening(); // First, stop listening so we can talk  radio.write( &got\_time, sizeof(unsigned long) ); // Send the final one back.  radio.startListening(); // Now, resume listening so we catch the next packets.  Serial.print(F("Sent response "));  Serial.println(got\_time);  }  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Change Roles via Serial Commands \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  if ( Serial.available() )  {  char c = toupper(Serial.read());  if ( c == 'T' && role == 0 ){  Serial.println(F("\*\*\* CHANGING TO TRANSMIT ROLE -- PRESS 'R' TO SWITCH BACK"));  role = 1; // Become the primary transmitter (ping out)    }else  if ( c == 'R' && role == 1 ){  Serial.println(F("\*\*\* CHANGING TO RECEIVE ROLE -- PRESS 'T' TO SWITCH BACK"));  role = 0; // Become the primary receiver (pong back)  radio.startListening();    }  }  } // Loop |

Draai op twee verschillende laptops en stel via seriële monitor de ene in op T(ransmit) en de andere op R(eceive)

## Nogmaals op 1 laptop



## Onderzoeksdeelvragen

**Onderzoek de doorvoersnelheid en bepaal de delay van een Arduino-Pi-verbinding met NRF24’s.**

In de diverse datasheets van de NRF24 worden theoretische maxima gegeven voor de maximale afstand tussen twee transceivers en de ondervonden delay.

Maak een grafiek aan de hand van uitgevoerde metingen van de maximale doorvoersnelheid en van de delay van een Pi- transceiver– transceiver– Arduino opstelling uitgezet tegen de afstand tussen de Pi-transceiver enerzijds en de Arduino-transceiver anderzijds.

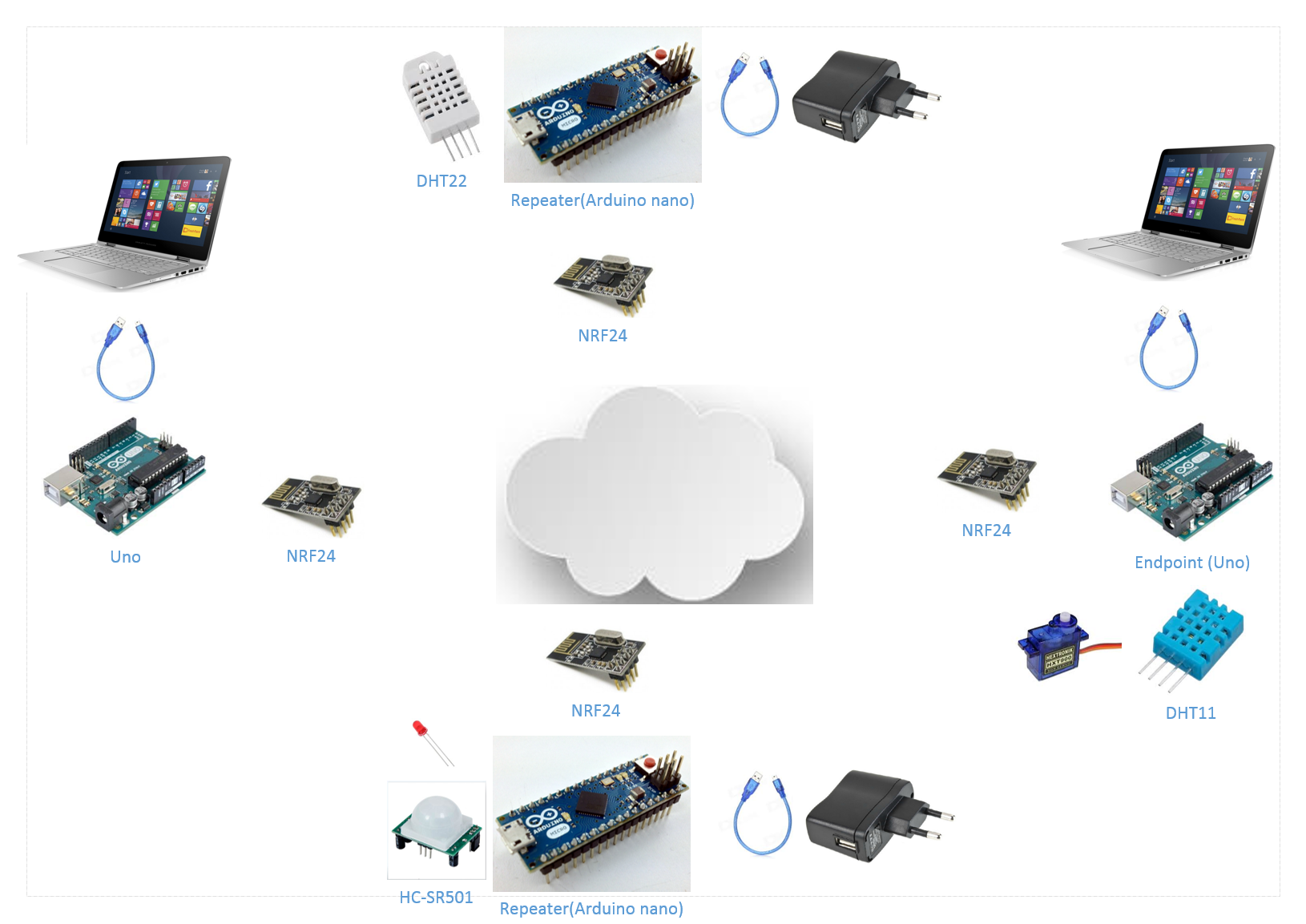
Voeg zelf een simplex applicatielaag toe d.w.z. je kunt fouten ontdekken op de applicatielaag, maar er zijn geen teruggestuurde bevestigingsberichten en geen retransmissies op grond van timeouts.

# Onderzoek netwerkarchitectuur

## Meetopstelling

We gaan onderstaande opstelling bouwen met:

* Gateway
  + Arduino Nano
  + Communiceert serieel met laptop/Pi via USB
  + Communiceert met draadloos netwerk via NRF24
  + Geen sensoren/actuatoren (in MySensors versie 1.5)
  + Nooit in slaapstand
* Repeater
  + Arduino Nano of Arduino mini
  + Communiceert met draadloos netwerk via NRF24
  + Een of meer sensoren
  + Een of meer actuatoren
  + Nooit in slaapstand
* End point
  + Arduino Nano of Arduino mini
  + Communiceert met draadloos netwerk via NRF24
  + Een of meer sensoren
  + Een of meer actuatoren
  + Kan in slaapstand



Voor de routers en endpoints kan er in het laboratorium gekozen worden voor een Arduino Uno (uit de ESD-kit in de propedeuse), een Arduino Pro mini of een Arduino Nano. Het gebruik van andere Arduino’s is natuurlijk ook mogelijk. Voor de gateway is het verstandiger een Arduino Uno of Nano te kiezen vanwege de seriële verbinding met de laptop, maar een Arduino mini met Ethernet-adapter kan ook.

<http://www.mysensors.org/about/arduino>

## Gebruik Pi

<http://www.hobbytronics.co.uk/raspberry-pi-serial-port>

## Software mesh netwerk

Voor de software gebruiken we een kant en klare library van MySensors.org

## Wissen EEPROM

De Arduino’s in het laboratorium kunnen al eerder gebruikt zijn en er kan dus informatie in de EEPROM zijn achtergebleven. Dit kan de experimenten verstoren. Daarom is het verstandig deze inhoud te wissen:

<https://www.mysensors.org/build/debug>

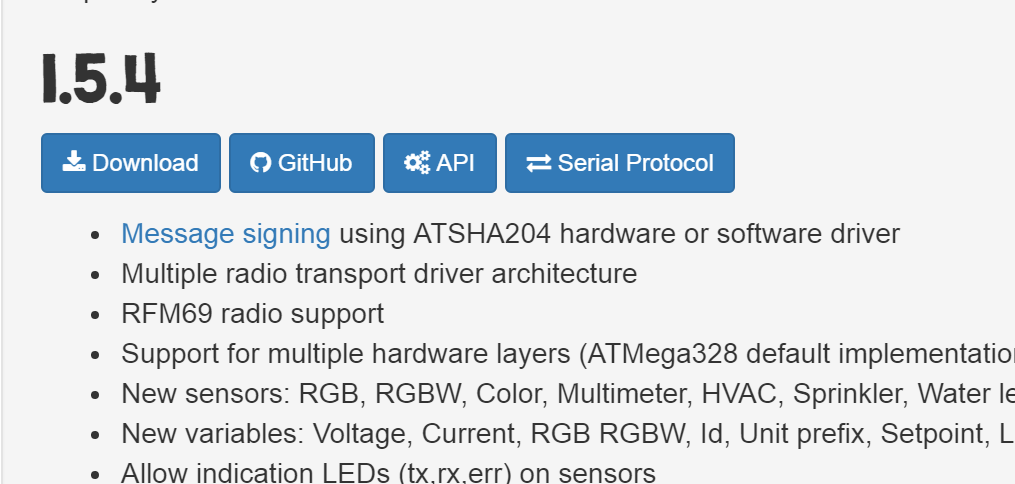
## Aansluiten gateway

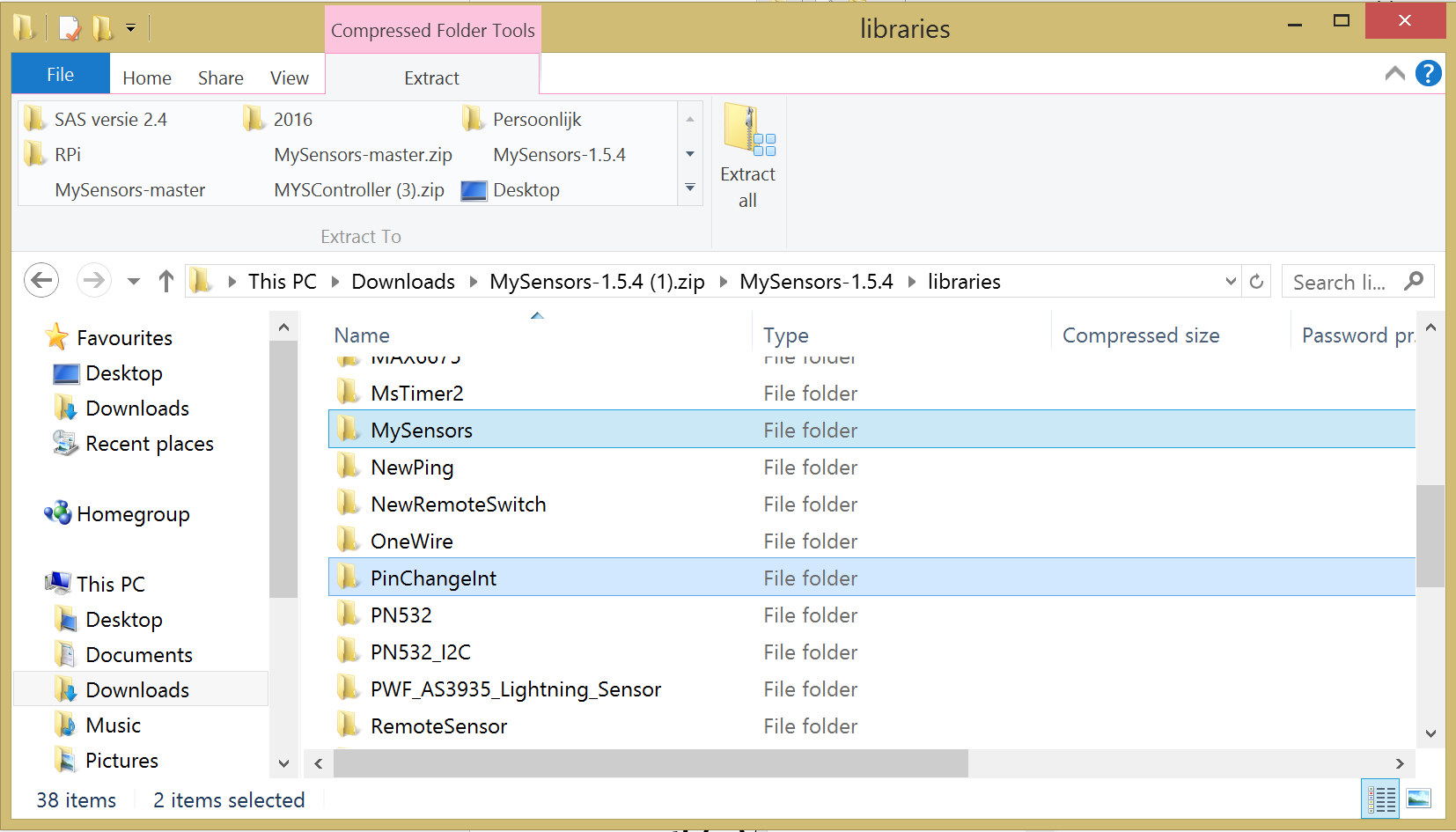
<https://www.mysensors.org/build/serial_gateway>

Dit is identiek aan het vorige hoofdstuk.

## Installeren MySensors library

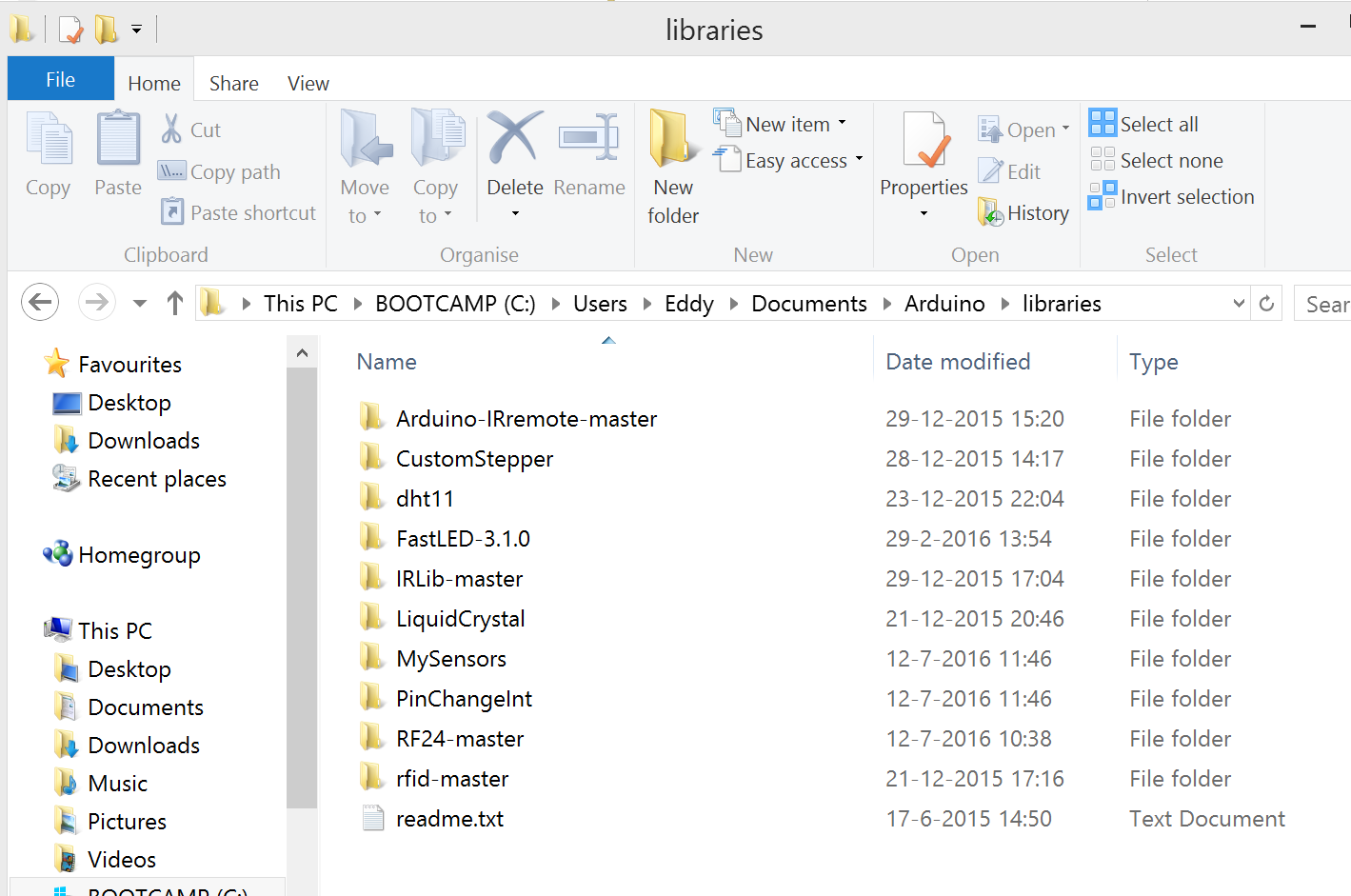
Op dit moment maakt MySensors een grote transitie door van versie 1.5 naar 2.0 In dit document zijn we voorlopig uitgegaan van 1.5



Download release 1.5.4, open het ZIP-bestand 

En kopieer alleen de MySensors en PinChangeInt mappen naar

C:\Users\Eddy\Documents\Arduino\libraries

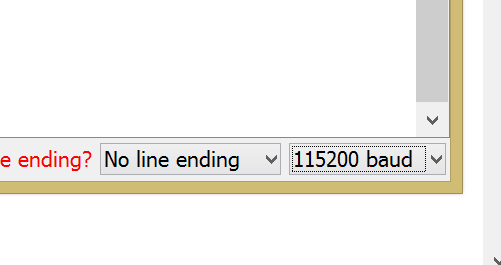


En vergeet niet de Arduino-IDE te sluiten en te openen.

## Gateway ‘programmeren’

Open uit de Arduino-IDE voorbeelden: seriële gateway.

Upload en open serial monitor op 115200 bps:



Na downloaden sketch:

Output 1.5:



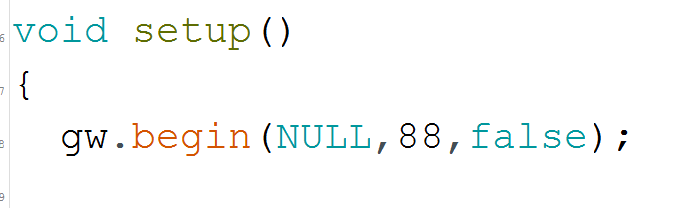
## Installeren end point

<http://www.mysensors.org/download/sensor_api_15>

Wij kozen voor de CHILD\_ID de waarde 77 te geven. Dit maakt het makkelijker straks de boodschappen vanuit dit end-point te herkennen



Geef in eerste instantie het eind-point een vast ID b.v. 88:



## MySensors seriële protocol

<http://www.mysensors.org/download/serial_api_15>

## Bestuderen debug-informatie

Bekijk de debug-informatie vanuit de gateway en vanuit het endpoint.

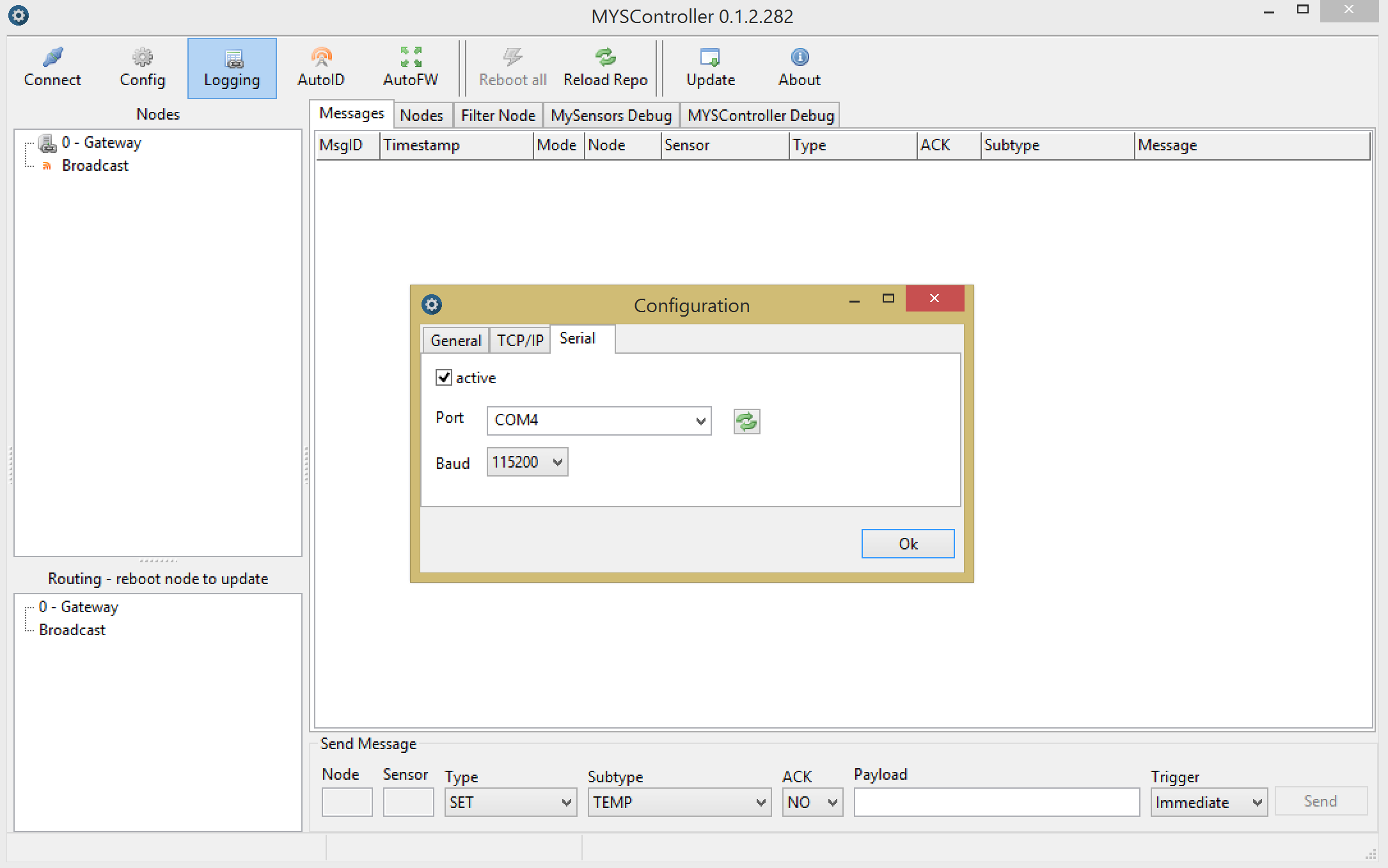
<https://www.mysensors.org/build/debug>

## Sniffen netwerkverkeer

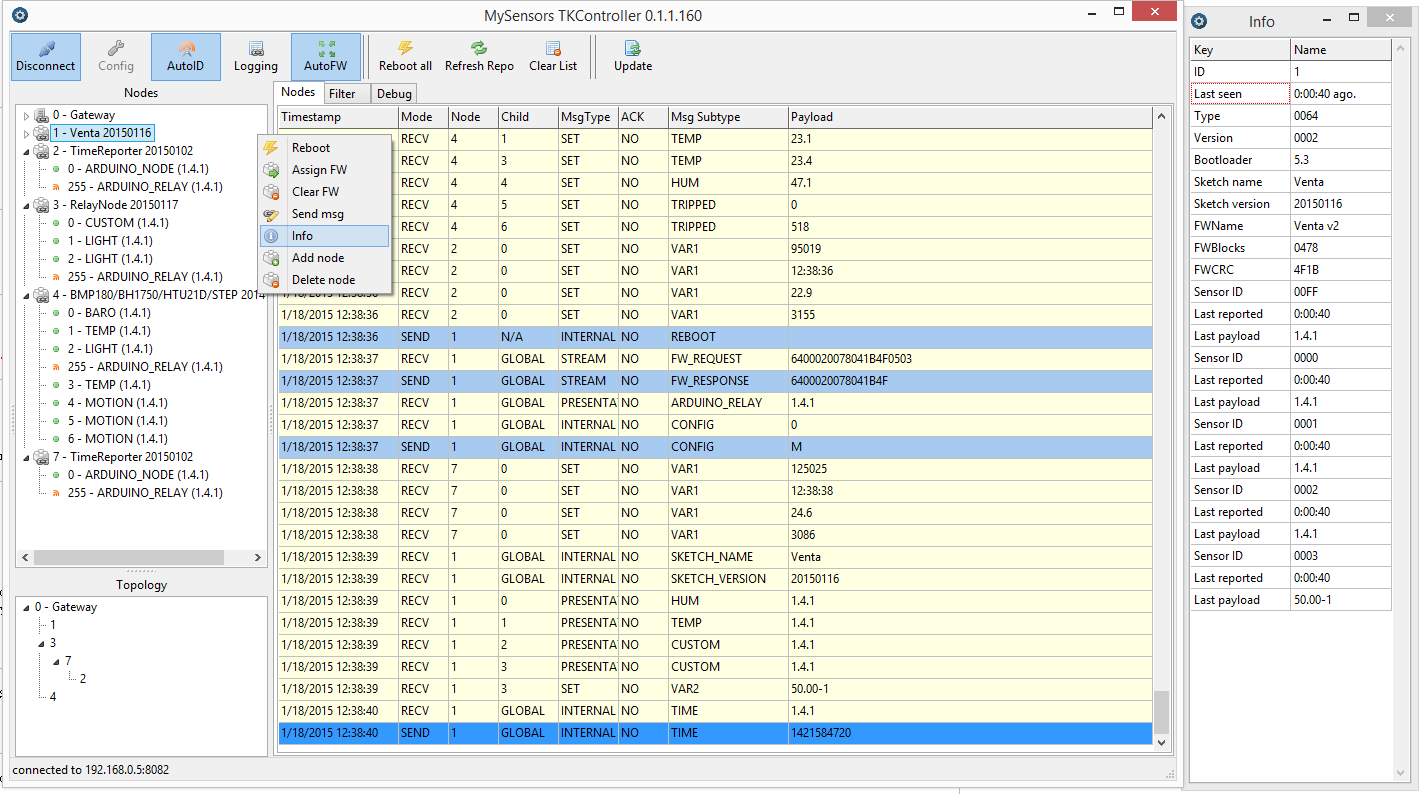
Hiervoor gebruiken we een tool van MySensors.org nl. de MYScontroller. Draait onder OS-X met Windows virtueel onder Parallels; vergeet niet de speciale drivers voor de CH340 te installeren.

**Belangrijk bij het installeren van de gateway: in de Arduino-code moet DEBUG uitgeschakeld worden anders toont het topology-venster niets!**

Stel de seriële poort in:



En druk op connect:



## Installeren routers

<http://www.mysensors.org/download/sensor_api_15#create-repeating-nodes>

## Onderzoeksvragen

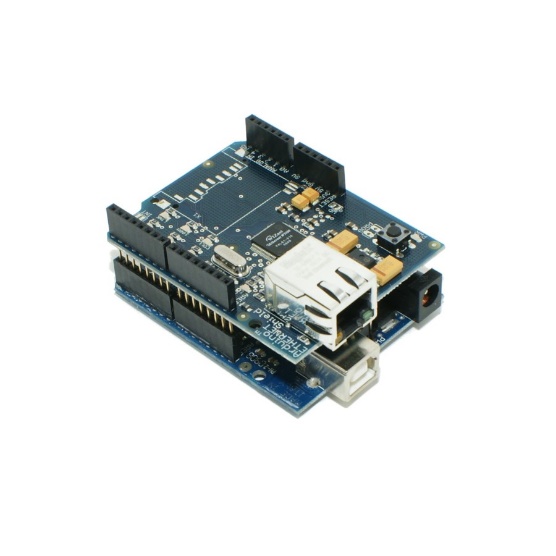
Stel zelf minstens 5 onderzoeksdeelvragen op en onderzoek deze m.b.v. de opstelling. Bespreek deze met de docent alvorens een laboratoriumonderzoek uit te voeren.

## Facultatieve opdracht

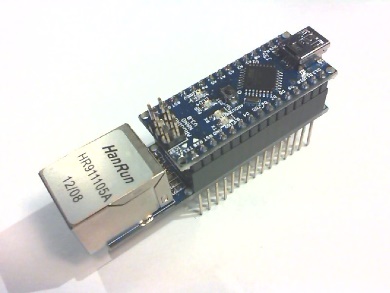
Probeer een endpoint telkens 1 minuut in slaapstand te brengen.

## Facultatieve opdrachten: Ethernet gateway

Vervang de seriële gateway door een gateway via Ethernet. Neem een Uno met Ethernet-shield:



Of neem een Arduino Nano met een ENC28J60 Ethernet shield:



Onderzoek welke opdrachten er via de Ethernet-verbinding worden verzonden en of dit al dan niet overeenkomt met een gateway via de seriële verbinding.

## Facultatieve opdracht

Maak een kleine controller in Python, die gegeven een openende deur of bewegende persoon een lamp gedurende 5 sec. inschakelt.

## Faculatieve opdracht

Maak een kleine MySensors – controller, die gegeven de melding van een nieuwe sensormodule dynamisch een nieuwe ID toekent.

## Facultatieve opdracht: Xbee

Voor studenten, die meer verdieping zoeken, in de centrale laboratoriumdoos liggen Xbee’s met adapter.