# Alarmsysteem via Radio communicatie

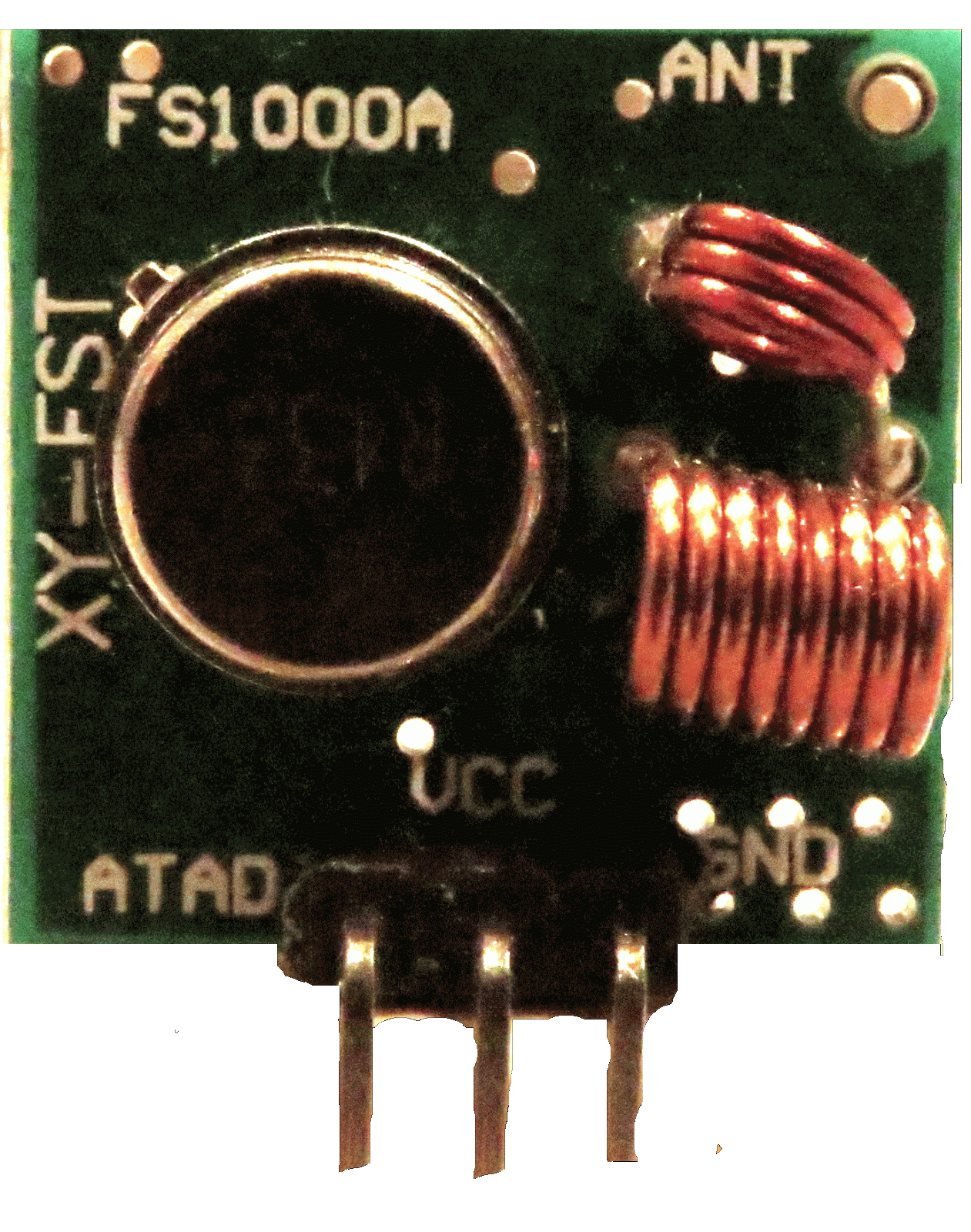
[Datasheet](http://www.mantech.co.za/Datasheets/Products/433Mhz_RF-TX&RX.pdf)

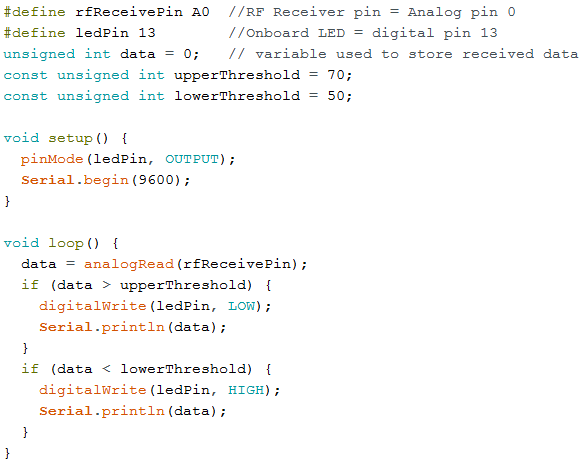
**Test**

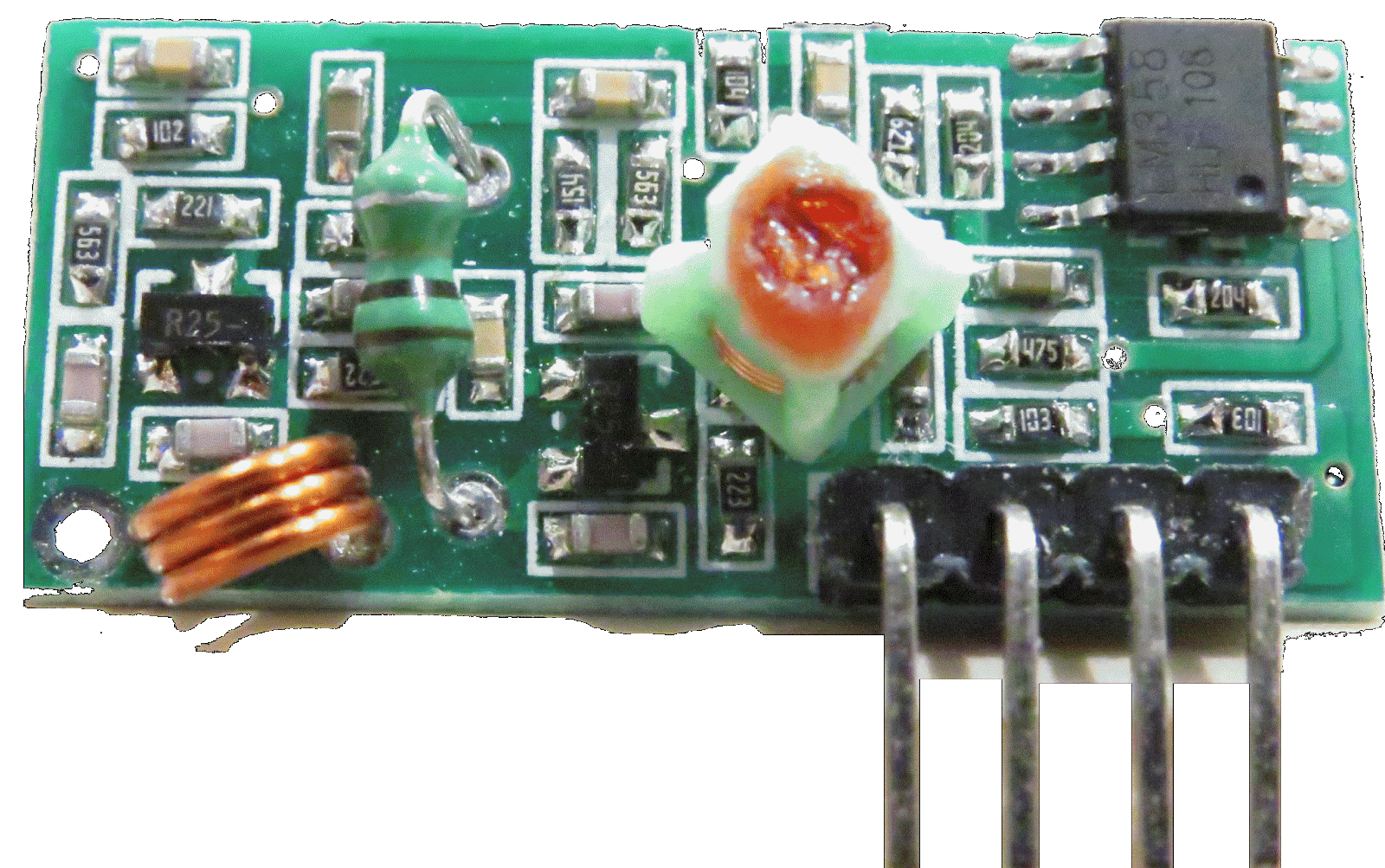
Bij het testen hebben we een simpel programma geschreven, waarbij de Transmitter een laag en een hoog signaal uitstuurt met een delay. De Receiver wordt analoog uitgelezen.



**Transmitter**

****



**Receiver**

**Dit is de data die we ontvingen met de Receiver**

2

2

1

2

2

2

2

1

249

767

767

767

767

767

767

767

767

766

767

767

767

767

767

767

767

767

767

1

1

1

1

767

767

767

767

767

767

767

766

766

766

767

767

767

767

767

767

767

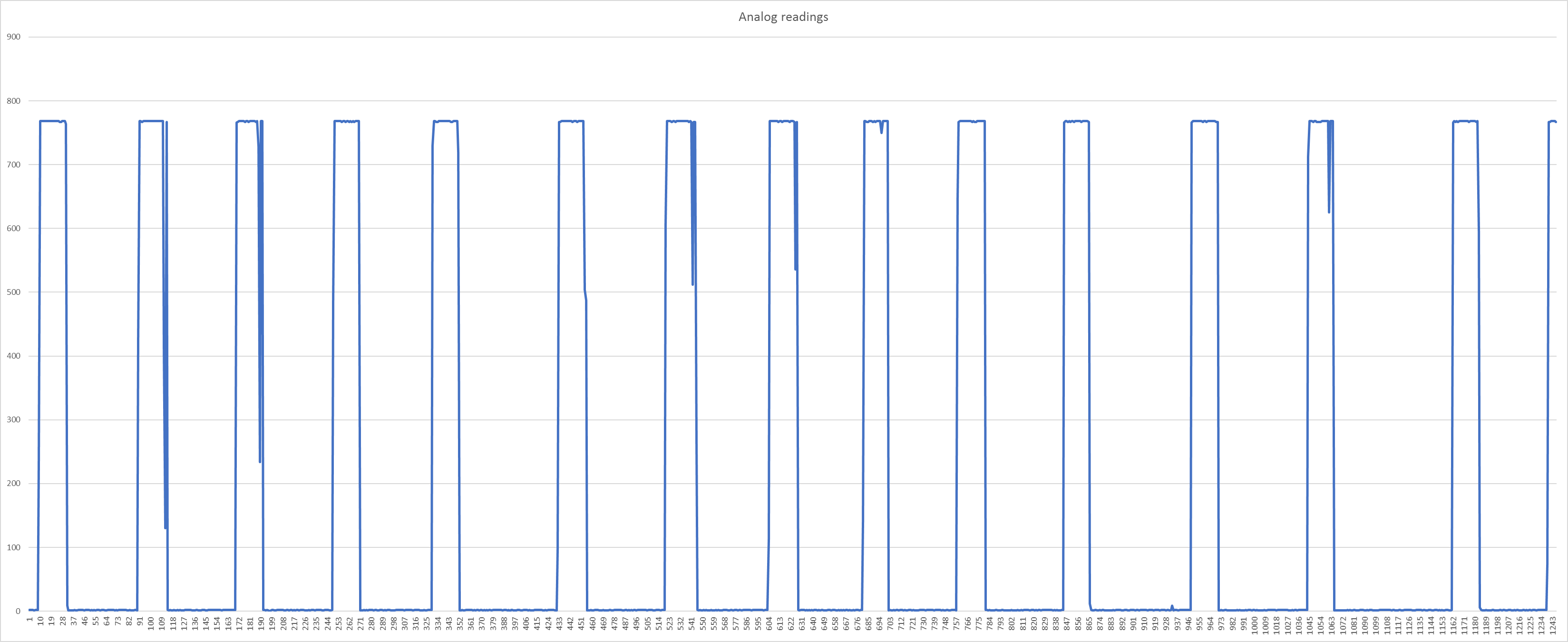
767

Wanneer de cijfers 2 en 1 weergegeven worden, werd er niks met de button gedaan. Wanneer de knop ingedrukt werd, werd het transmitter signaal op HIGH gezet.

Transmitter HIGH = Receiver 249, 767, 766

Transmitter LOW = Receiver 2, 1

**Grafiek van de ontvangen data**



Grafiek 1: analog readings

We zien dat er tijdens het versturen van data dat er een korte delay op zit, de precieze meting is lastig omdat we beide Arduino’s tegelijkertijd aan moeten zetten. We schatten een delay van minder dan één seconde.

Bij het gebruik van 3V kregen we geen goede data door, de data was inconsistent. De data hierboven is op de 5V pin verkregen.

Het valt op dat wanneer we het transmittersignaal continue op hoog zetten, de receiver maar een keer een hoog signaal geeft. Het lijkt dat de receiver niet continue een signaal kan ontvangen maar met tussenposen, zoals in grafiek 1 waarbij we de knop één keer lang indrukte.

We hebben een library [RadioHead](http://www.airspayce.com/mikem/arduino/RadioHead/RadioHead-1.59.zip) gebruikt, die ons voorziet van functies die een bericht door radiocommunicatie verstuurd. We kunnen nu berichten versturen die ontvangen worden als Hexadecimale getallen. We wilden de code zelf maken, maar dat is voor nu nog te complex. Als we tijd overhebben kunnen we hier meer tijd in besteden.

In de receiver staat driver.printBuffer("Got:", buf, buflen); deze methode hebben we opgezocht in de library en staat in de RHGenericDriver.cpp file. Deze konden we aanpassen zodat we strings doorgestuurd krijgen.



Hier hebben we gekeken waar de uint8\_t\* buf uit bestaat. Wanneer we een char pointer array ingeven wordt die in de send functie (uit de RadioHead Class) gecast naar een byte pointer array.

We gaan kijken om zelf berichten versturen met een byte array.

Byte arr[] = {B00010001};

Serial.println((char)arr[1]);

We maken hier een byte array met 1 binair getal, hierna casten we de arr naar een character, waardoor er een letter in de Seriële monitor verschijnt.