**Documentatie iRobot Roomba 600 serie**

**Tom Koevoets**

Avans Hogeschool, Breda

Communication & Multimedia Design

**Patricia Nagtzaam**

Avans Hogeschool, Breda

Communication & Multimedia Design

iRobot Roomba 600 documentation (open interface)

<https://www.irobotweb.com/~/media/MainSite/PDFs/About/STEM/Create/iRobot_Roomba_600_Open_Interface_Spec.pdf>

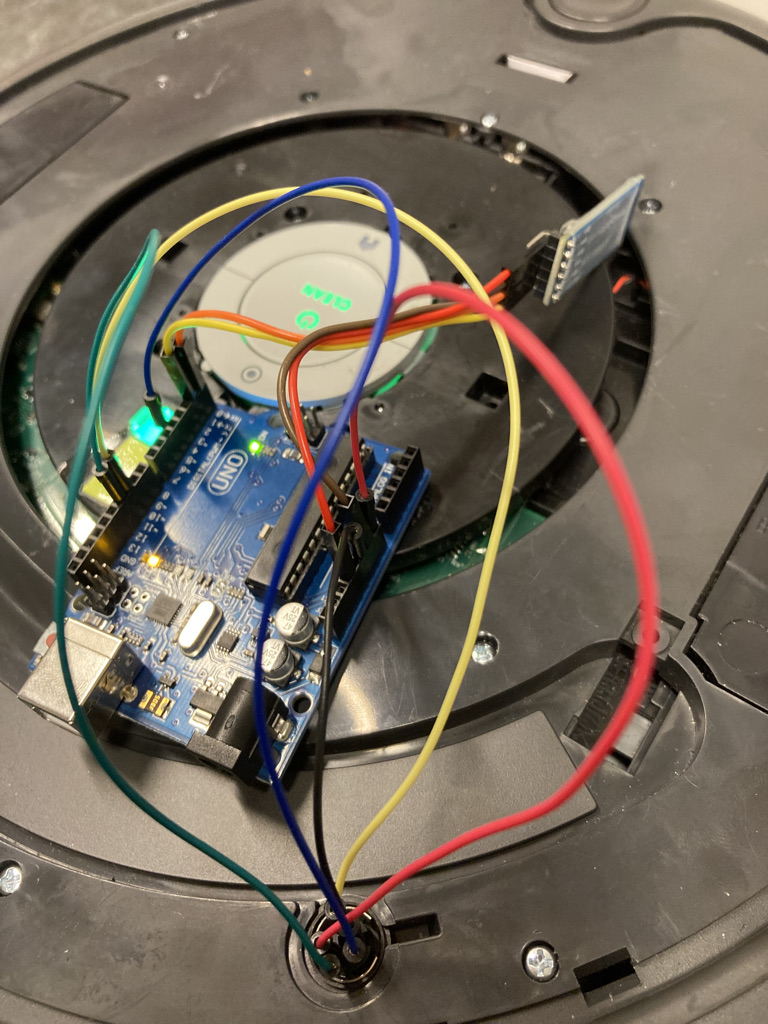
Controlling a Roomba Robot with Arduino and Android

[**https://www.instructables.com/Controlling-a-Roomba-Robot-With-Arduino-and-Androi/**](https://www.instructables.com/Controlling-a-Roomba-Robot-With-Arduino-and-Androi/)

RoombaDefines.h geraadpleegd van:

<https://github.com/simoneluconi/Roomba_ESP8266_Ctrl/tree/master/Roomba_ESP8266_Ctrl>

Afbeelding met vloer, binnen

Automatisch gegenereerde beschrijving

**In het kort**

In het komende document geven we uitleg over hoe wij een iRobot Roomba serie 600 (698) hebben aangestuurd met C++ via Arduino en de 7-pin connector op de Roomba zelf.

**De basics**

Om een Roomba via je telefoon te kunnen besturen moet je hem kunnen verbinden met de plaatselijke wifi. Bij ons op Avans is dat aardig lastig, maar via het medialab konden we het MAC-adres van de Roomba in het systeem zetten: (ziet er zo ongeveer uit; 00:E9:3A:7A:0C:C9) en hebben we hem op AvansWlan aangesloten. Hierdoor is deze iRobot Roomba te gebruiken overal op Avans, je hoeft alleen maar te verbinden via het wifi netwerk. Je hoeft hiervoor alleen maar een account te maken en de Roomba hieraan toe te voegen, je kan hem zelfs een naam geven.

Als je de Roomba hebt verbonden met je mobiele telefoon kan je hem normaal besturen en programma’s laten draaien vanuit de app. Wij hadden het probleem dat de Roomba niet opstartte als hij niet verbonden was met de Wi-fi. Als de Wi-fi wel meewerkt heb je een normaal werkende Roomba.

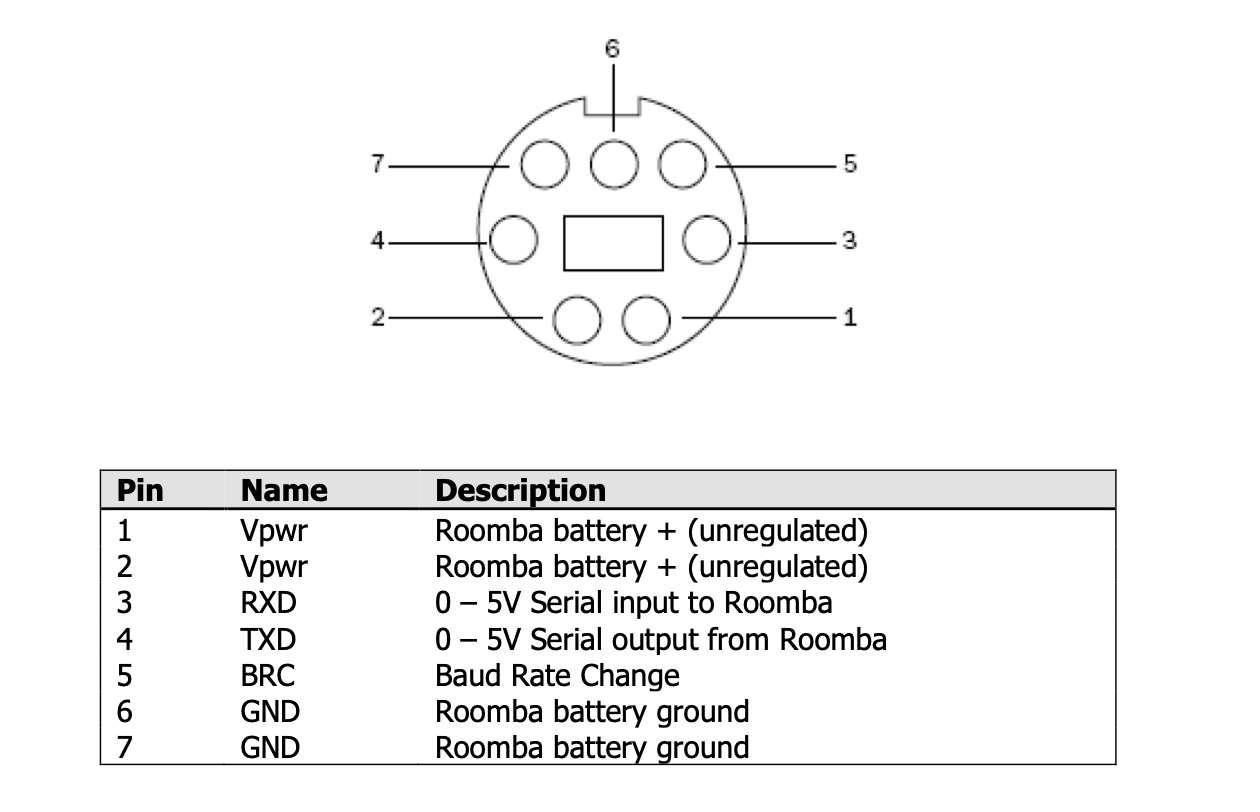
De controles en besturing worden duidelijk uitgelegd in de open interface die iRobot zelf publiceert van de Roomba (vooraan in het bestand gelinkt)

**Arduino verbinden met de Roomba**

Afbeelding met tekst, elektronica, schermafbeelding

Automatisch gegenereerde beschrijvingJe kan een Arduino heel simpel verbinden met de Roomba, er zit een female 7-pin connector onder de bovenste kap van de Roomba. Haal je deze kap eraf kan je makkelijk bij de connector.

Met deze afbeeldingen kan je een Arduino makkelijk verbinden met de Roomba. Hieronder zie je hoe de female 7-pin connector werkt. Je gebruikt 1 voor Power. LET OP dit is geen 5V maar unregulated, GEBRUIK hiervoor de vin pin op je Arduino. Anders doet je Arduino het niet lang meer. Pin 3 en 4 zijn voor Tx en Rx, als je data wilt uploaden naar je Arduino zal je deze uit de Roomba 7-pin connector moeten halen anders zal je een foutmelding krijgen.

Pin 5 is voor de Baud Rate, het is ook heel belangrijk dat als je een nieuwe code wilt uploaden naar de Arduino dat je de baudrate van de Roomba veranderd naar 19200 (de baudrate staat normaal op 115200), dit doe je door de clean-button op de Roomba 10 a 12 sec in te drukken als de Roomba uit staat tot je een aflopend deuntje hoort, doe je dit niet dan snapt de Roomba de code niet. Als laatste heb je Pin 6, dit is GND en deze kan je simpel verbinden met de GND.

**Arduino code**

Bij het schrijven van de code hebben we veel gebruik gemaakt van de open source documentation van iRobot zelf en een vergelijkbaar document waarmee een Roomba aangestuurd kan worden via Arduino, Bluetooth en Android.

De code waarmee de Roomba werkt is heel simpel, het bestaat namelijk uit een lijst van commands die je kan sturen naar de Roomba, elk van deze commands is uitgebreid uitgelegd in de documentatie. Hiermee kan je zelf de complexiteit van de code bedenken.

Ook heb je een aantal #include nodig in je bestand; namelijk <SoftwareSerial.h> en “roombaDefines.h” die gelinkt is boven aan het bestand. Hiervoor is dit stukje code nodig, hiermee vertel je de Roomba wat de Rx en Tx pin zijn en hoe er gecommuniceerd kan worden met de Roomba.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Wij gebruiken de simpelste variant van deze code; opstarten, vooruit rijden, achteruit rijden, draaien en stoppen. Het makkelijkst is om hiervoor void functies te schrijven, die je daarna kan opvragen in de setup of loop.

Om de Roomba te laten starten gebruiken we Safe start, hiervoor stuur je eerst de Roomba start(128) en vervolgens safe Mode (131), als je full control(132) wilt over de Roomba stopt hij niet meer als de Roomba van een trap afrijdt of word opgetild.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Nu de Roomba is opgestart kan je de volgende functies starten, wij gebruiken naar voren rijden, achteruit rijden en halt. Ze komen allemaal uit de code drive(137) van Roomba. Het nadeel is dat ze alleen maar hexadecimal lezen. Voor de Velocity kan dit maximaal 255, dus 200 is goed zodat je de Roomba langer kan laten rijden. Via een online convertor konden we de waardes vertalen naar Hex, de volgorde ervan is lastig in te vullen, je splitst het nummer op over high en low velocity byte achter de 0x.

Als je de code(137) plus 4 data doorstuurt kan je de Roomba vooruit of achteruit laten rijden, halt is daar de makkelijke variant van “0x00” staat gelijk aan “0”, hierdoor stoppen alle rij functies en stopt de Roomba.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Nu moet het zijn gelukt om de Roomba te Safe starten, vooruit en achteruit laten rijden.

Om de Roomba te kunnen laten draaien op zijn plaats gebruiken we drive direct(145), er zijn meerder manieren mogelijk om de Roomba te laten draaien, deze staan allemaal duidelijk vermeld in de iRobot Documentatie.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Met driveWheels() kun je bepalen welke kan de Roomba op moet draaien, dit kan in een range van -500 tot 500. Dit kun je aansturen door de 2 waardes te veranderen;

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Door hier een bepaalde delay aan te geven kan je precies bepalen hoe ver de Roomba mag draaien voordat hij weer vooruit of achteruit rijdt.

**Connect Arduino met HC-05 Bluetooth module**

Een uitleg over hoe je de Roomba kan aansturen via een Android. Dit hebben we samen gedaan met een HC-05 bluetooth en een oude Samsung.

Afbeelding met tekst, elektronica

Automatisch gegenereerde beschrijvingOm te beginnen had ik op de Android een applicatie geïnstalleerd “Arduino bluetooth controller” waarmee ik over bluetooth tekens kon sturen naar mijn Arduino. Deze tekens; A, V, L, R, 0 en 1 had ik gekoppeld aan functies. Zodat we de Roomba op afstand met Bluetooth konden besturen.

Om de HC-05 te verbinden kun je dit

schema hieronder volgen:

Als je de HC-05 hebt geïnstalleerd kan je deze uitlezen met een string + char.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Met deze 2 functies werkt de Bluetooth module, “inputString” is de data die word verzonden vanuit de Android. Door functies te koppelen aan deze “tekens” is het mogelijk om de Roomba te besturen via een simpele contoller.

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Afbeelding met tekst

Automatisch gegenereerde beschrijvingHet enige wat je dan nodig hebt zijn een heel aantal what-ifs, dat zodra er een A, V, L, R, 0 of 1 wordt verstuurd dat de Roomba weet wat hij moet doen.

De uiteindelijke code zorgt er dan voor dat de Roomba op afstand bestuurd kan worden met een Android via Arduino en een HC-05 bluetooth module.

**Code C++ Controller Arduino & Android**

#include <SoftwareSerial.h>

#include "roombaDefines.h"

//Roomba Create2 connection

int rxPinRoomba = 10;

int txPinRoomba = 11;

SoftwareSerial Roomba(rxPinRoomba,txPinRoomba);

char junk;

String inputString="";

void startSafe() {

Roomba.write(128); //start

Roomba.write(131); //SAFE mode

}

void goForward() {

Roomba.write(137); //Drive

Roomba.write((byte)0x00); //0x00c8 == 200 velocitye high byte

Roomba.write(0xc8); //Velocity low byte

Roomba.write(0x80); //Radius high byte

Roomba.write((byte)0x00); //Radius low byte

}

void goBackward() {

Roomba.write(137); //Drive

Roomba.write(0xff); //0xff38 == -200 velocitye high byte

Roomba.write(0x38); //Velocity low byte

Roomba.write(0x80); //Radius high byte

Roomba.write((byte)0x00); //Radius low byte

}

void halt() {

byte j = 0x00;

Roomba.write(137);

Roomba.write(j);

Roomba.write(j);

Roomba.write(j);

Roomba.write(j);

}

void driveWheels(int right, int left) {

clamp(right, -500, 500);

clamp(left, -500, 500);

Roomba.write(145);

Roomba.write(right >> 8);

Roomba.write(right);

Roomba.write(left >> 8);

Roomba.write(left);

}

void setup() // run once, when the sketch starts

{

Serial.begin(9600); // set the baud rate to 9600, same should be of your Serial Monitor

Roomba.begin(19200);

pinMode(ddPin, OUTPUT);

}

void loop() {

//READOUT incomming bluetooth message, then do task

if(Serial.available()){

while(Serial.available())

{

char inChar = (char)Serial.read(); //read the input

inputString += inChar; //make a string of the characters coming on serial

}

Serial.println(inputString);

while (Serial.available() > 0)

{ junk = Serial.read() ; } // clear the serial buffer

if(inputString == "R"){ //RECHTS

driveWheels(-150,150); //range -500/500

} else if(inputString == "A"){ //ACHTERUIT

goBackward();

} else if(inputString =="V"){ // VOORUIT

goForward();

} else if(inputString == "L"){ //LINKS

driveWheels(150,-150); //range -500/500

} else if(inputString == "0"){ //turn OFF

halt();

} else if(inputString == "1"){ //turn ON

startSafe();

}

inputString = "";

}

}