Arduino verstehen und anwenden

Teil 26: Drahtlos messen und steuern – Funkmodule machen es möglich

Drahtlose Kommunikation

In diesem Beitrag sollen verschiedene Methoden vorgestellt werden, welche die drahtlose Kommunikation zwischen zwei Arduino-Boards ermöglichen. Es werden drei verschiedene Verfahren eingesetzt:

- Serielle Funkstrecke mit 433-MHz-Modulen
- Module der RFM-Serie im 433- oder 868-MHz-Bereich
- nRF24L01-Module für das 2,4-GHz-Band

Mithilfe hochfrequenter Signale können Daten, Schaltbefehle oder Messwerte über größere Entfernungen drahtlos übertragen werden. Allerdings ist sowohl die Erzeugung als auch der Empfang von Hochfrequenz relativ aufwendig.

Häufig war der Eigenbau von Radioempfängern in früheren Jahren der Einstieg in das weite Feld der Elektronik. Mittelwellenempfänger waren noch recht einfach zu realisieren, praxistaugliche UKW-Radios galten bereits als Meisterstück der Hobbyanwendung. Dies ergibt sich aus der Tatsache, dass bei höheren Frequenzen ab einigen Megahertz zunehmend Probleme auftreten. Selbst minimale Kapazitäten oder Induktivitäten zeigen nun ihre störenden Einflüsse. Verbindungen von wenigen Zentimetern Länge können bereits die Empfangseigenschaften einer Schaltung stark beeinflussen. Ab etwa 10 MHz wird die Hochfrequenztechnik so geradezu zum Hexenwerk.

Der Eigenbau von Sendern und Empfängern in Megahertz-Bereich erfordert schon umfangreiches Spezialwissen. Ein Aufbau auf einem Breadboard ist aufgrund der inhärenten Streukapazitäten und -induktivitäten meist von vornherein zum Scheitern verurteilt. Zudem ist das Senden von elektromagnetischen Wellen stark reglementiert. Sogenannte „Schwarzsender" sind in den meisten Ländern der Welt streng verboten und der Betreiber muss mit hohen Strafen rechnen.

Seit einigen Jahren existieren aber verschiedene Möglichkeiten, die auch dem hochfrequenztechnischen Laien Funkübertragungen auf vollkommen legalem Weg gestatten. Verschiedene Module für eng begrenzte Frequenzbänder enthalten die gesamte Hochfrequenzelektronik inklusive Antenne, Sender und Modulationseinheit. Es muss nur noch das niederfrequente Datensignal am Sender eingespeist werden, dann wird dieses drahtlos an einen zugehörigen Empfänger übertragen. Dort wird es demoduliert und steht in hoher Qualität wieder zur Verfügung. Auf diese Weise wird die kabellose Datenübertragung fast zum Kinderspiel und einer auf Funktechnik basierenden Heimautomatisierung stehen keine hochfrequenztechnischen Hürden mehr im Weg.

Freie Frequenzbänder

Die für Privatanwender freigegebenen Frequenzen sind als sogenannte ISM-Bänder bekannt. ISM steht dabei für Industrial, Scientific and Medical, also Industrie, Wissenschaft und Medizin. Ein wichtiger Teilbereich ist das 2,4-GHz-Band, für welches viele preisgünstige Module verfügbar sind.

Die folgenden Frequenzbereiche wurden für allgemeine Anwendungen freigegeben:

- 13,553 MHz bis 13,567 MHz
- 26,957 MHz bis 27,283 MHz
- 40,66 MHz bis 40,70 MHz
- 150 MHz
- 433,05 MHz bis 434,79 MHz
- 2400 MHz bis 2500 MHz

Die ISM-Frequenzbänder werden von zahlreichen Funkanwendungen genutzt. Hierzu gehören:

- Alarmfunk
- Baby-Überwachungsanlagen
- Audio- und Funkmikrofon-Anwendungen
- Funk-Bewegungsmelder
- Hörhilfen
- Fernsteuerungen im Modellbau

Wichtiger Hinweis:

Die Verwendung von Sende- und Empfangsanlagen liegt ausschließlich im Verantwortungsbereich des Betreibers. Dieser ist für die Einhaltung der rechtlichen Bestimmungen in seinem Land selbst verantwortlich.
Eine Haftung von Verlag oder Autor ist ausgeschlossen.

Serielle Funkbrücke im 433-MHz-Band

„VirtualWire" ist eine Arduino-Bibliothek, die es erlaubt, kurze Nachrichten, Messdaten oder Ähnliches drahtlos zu versenden. Sie unterstützt eine Reihe von preiswerten Funksender- bzw. Funkempfänger-Paaren. Die einzelnen Übertragungspakete enthalten eine Trainingspräambel, die Nachricht selbst sowie eine CRC-Prüfsumme.

Alternativ wäre es natürlich auch möglich, die Arduino-UART direkt an Sender und Empfänger anzuschließen. Dann stünden aber weder die Trainingssequenz noch die CRC-Routine zur Verfügung. Dies hätte eine deutlich schlechtere Reichweite bzw. eine wesentlich höhere Fehlerquote zur Folge.

Die VirtualWire-Bibliothek kann unter
https://github.com/sparkfun/RF_Links/tree/master/Firmware/Arduino/libraries/VirtualWire
aus dem Internet geladen werden.

Nach der Installation der Bibliothek kann mit dem folgenden Sketch eine Nachricht gesendet werden:

```
// simple_transmitter_433_MHz.ino
#include <VirtualWire.h>

const int transmit_pin = 2;
const int led_pin = 13;
char msg[15];

void setup()
{   vw_set_tx_pin(transmit_pin);
    vw_setup(2000);  // Bits per sec
    pinMode(led_pin, OUTPUT);
}

byte count = 1;

void loop()
{ char msg[]="test test test";
  digitalWrite(led_pin, HIGH); // flash LED for transmit
  vw_send((uint8_t *)msg, strlen(msg));
  vw_wait_tx();
  digitalWrite(led_pin, LOW);
  delay(1000);
}
```

Der zugehörige Empfänger-Sketch sieht so aus:

```
// simple_receiver_433_MHz.ino
#include <VirtualWire.h>

const int receive_pin = 2;

void setup()
{   Serial.begin(9600);
    Serial.println("Rx setup");
    vw_set_rx_pin(receive_pin);
    vw_setup(2000);        // Bits per sec
    vw_rx_start();
}

void loop()
{   uint8_t buf[VW_MAX_MESSAGE_LEN];
    uint8_t buflen = VW_MAX_MESSAGE_LEN;
    if (vw_get_message(buf, &buflen))
    {Serial.print("received message: ");
for (int i = 0; i < buflen; i++) Serial.print(char(buf[i]));
          Serial.println();
    }
}
```

Die Hardware-Verdrahtung ist denkbar einfach. Die Module müssen lediglich mit einer Versorgungsspannung von jeweils 5 V versehen werden. Der Daten-Pin (Tx-data für den Sender und Rx-data für den Empfänger) ist in beiden Fällen mit dem I/O-Pin D02 eines Arduinos zu verbinden.

Bild 1 zeigt ein Aufbaubeispiel für einen Handsender mit einem 433-MHz-Sendemodul. Ein zugehöriges Empfangsmodul, hier vom Typ Velleman (Bestell-Nr. CR-12 90 57, siehe Abschnitt „Empfohlene Produkte" am Ende des Artikels) ist in Bild 2 zu sehen. In Bild 3 ist die serielle Datenübertragung auf einem Oszilloskop dargestellt. Man erkennt, wie zunächst die Trainingssequenz übertragen wird und dann die eigentlichen Daten folgen.

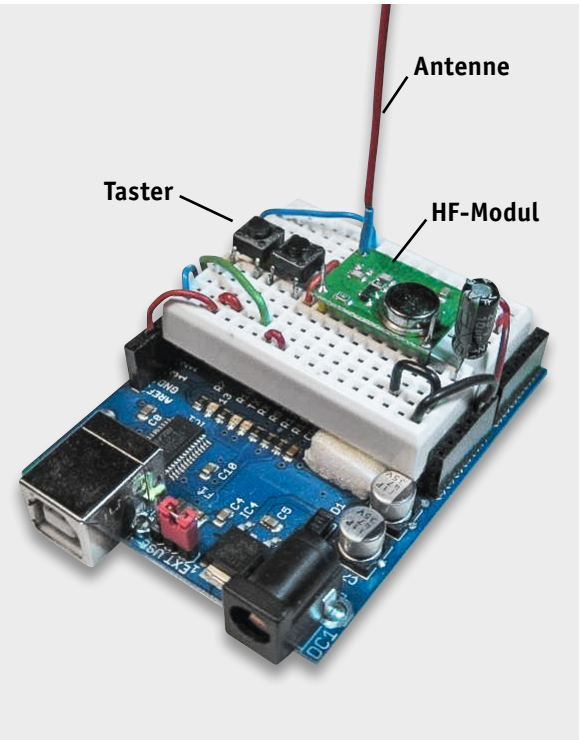

Bild 1: Arduino-Handsender mit 433-MHz-Sendemodul

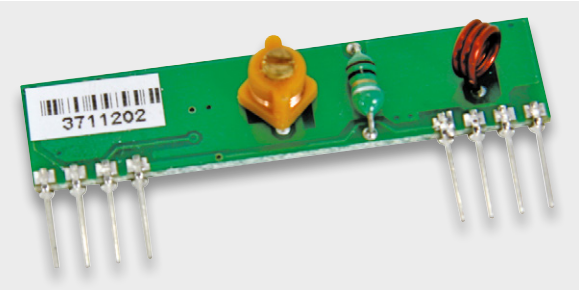

Bild 2: Empfänger-Modul (Typ Velleman)

Mit diesem Verfahren werden nur relativ kleine Reichweiten erzielt. Zudem zeigt die Methode nur eine geringe Fehlertoleranz. Die folgenden Varianten liefern hier deutlich bessere Ergebnisse.

Module der RFM-Serie

RFM12(b)-Module zeichnen sich sowohl durch ihren geringen Preis als auch durch ihre hohe Leistungsfähigkeit aus. Für unter € 5,– erhält man ein komplettes Modul mit SPI-Interface, mit dem Entfernungen von bis zu 100 m überbrückt werden können. Im Vergleich zu anderen Funkstandards wie Xbee oder Bluetooth kann man mit den RFM-Modulen also sehr preisgünstige Funkbrücken aufbauen.

Die aktuelle Version der RFM-Serie ist das Modul RFM12b. Dieses zeichnet sich durch die folgenden Leistungsmerkmale aus.

- Betriebsspannung 2,2...3,8 V
- SPI-Schnittstelle
- Hohe Empfangsempfindlichkeit von -105 dBm
- Keine Abstimmung notwendig
- FSK-Modulation
- Hohe Datenrate von bis zu 115,2 kBit/s
- Automatische Antennenabstimmung

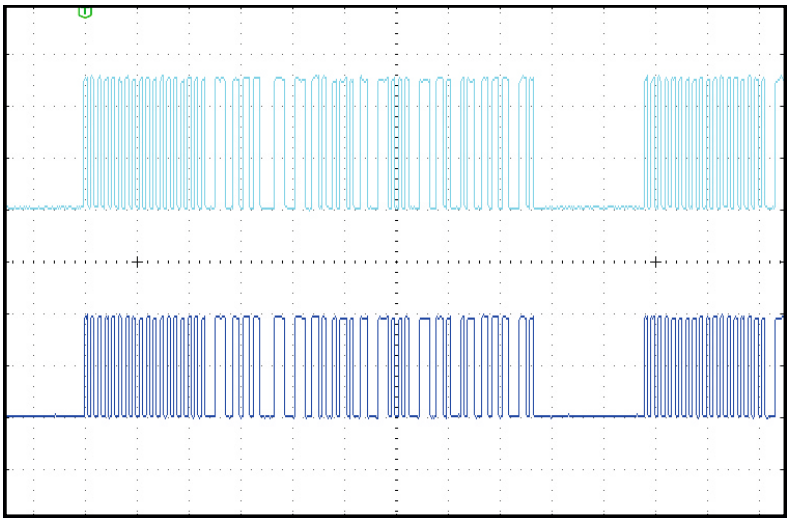

Bild 3: Sende- und Empfangssignal auf einem Oszilloskop

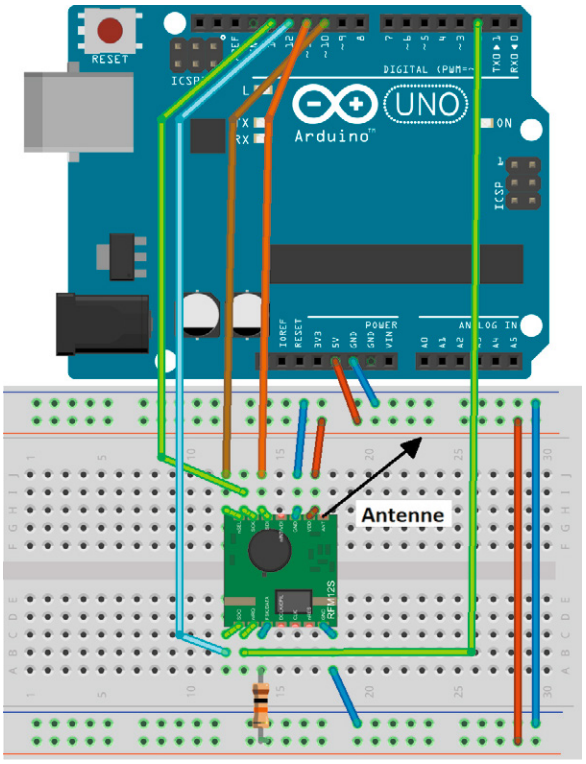

Bild 4: RFM12-Modul am Arduino UNO

- Programmierbare Empfänger-Bandbreite
- Geringe Stand-by-Stromaufnahme von weniger als 0,3 µA

Die Vorgängermodelle aus der RFM12-Reihe erfreuen sich jedoch weiterhin großer Beliebtheit, da sie in Arduino-Projekten sogar den Vorteil haben, dass sie mit 5 V betrieben werden können.

Für den Einsatz steht eine sehr hilfreiche Bibliothek zur Verfügung, sodass die Anwendung im Smart Home Bereich keine Probleme bereitet.

Die RF12-Library kann unter
https://github.com/jcw/jeelib
heruntergeladen werden.

Bild 4 zeigt den zugehörigen Hardware-Aufbau. Für einen Betrieb des Moduls sind also folgende Verbindungen notwendig:

- SDO an Pin D12 (MISO)
- nIRQ an Pin 2 (INT0)
- FSK an Vcc (via 10 kΩ)
- nSEL an Pin D10 (SS)
- SCK an Pin D13 (SCK)
- SDI an Pin D11 (MOSI)
- GND an Masse
- VDD an P5V

An den mit ANT bezeichneten Anschluss muss eine geeignete Antenne angeschlossen werden. Im einfachsten Fall kann das ein Stück Draht mit einer Länge von etwa 173 mm sein.

Dies entspricht bei der Funkfrequenz von 433 MHz einer Lambda-Viertel-Antenne:

$1/4*(c/f) = 173\ mm$

mit

$c = 3*10^8\ m/s$: *Lichtgeschwindigkeit*

$f = 433\ MHz = 433*10^6\ Hz$

Die Bezeichnungen MISO, MOSI und SCK deuten auf das SPI-Interface des Moduls hin. Dieses sollte für aufmerksame Leser der letzten Artikel dieser Serie kein Geheimnis mehr darstellen.

Dank der Library gestaltet sich auch die Softwareseite sehr einfach. Für die Senderseite kann der folgende Sketch eingesetzt werden:

```
#include <JeeLib.h>
int ADCvalue;

void setup()
{ pinMode(8, OUTPUT);
  Serial.begin(9600);
  Serial.println("Tx-Setup");
  // node 10, 433 MHz, RFM group 212
  rf12_initialize(10, RF12_433MHZ, 212);
}

void loop()
{ ADCvalue=analogRead(A0);
  rf12_sendStart(0, &ADCvalue, sizeof ADCvalue);
  Serial.print("Data sent: ");
  Serial.println(ADCvalue, DEC);
  digitalWrite(8, HIGH); delay(100);
  digitalWrite(8, LOW);  delay(100);
}
```

Er überträgt das am A0-Eingang des Arduinos anliegende Analogsignal an einen entsprechenden Empfänger. Damit können alle Arten von Analogsensor-Signalen wie etwa von Thermometern, Luxmetern o. Ä. via Funk übertragen werden. Für das hier verwendete RFM12-Modul muss das letzte Argument in der Funktion rf12_initialize 212 sein, für andere Module muss der Wert entsprechend abgeändert werden.

Das RFM12 ist explizit 5-V-tolerant. Daher werden für diese Version keine Pegelwandler benötigt. Für die neueren Module der RFM12b-Serie dagegen ist eine Versorgungsspannung von 3,3 V erforderlich. Auch die Signalpegel müssen auf diesen Wert ange-

passt werden. Im einfachsten Fall kann man dafür Spannungsteiler (z. B. 4,7–10 kΩ) einsetzen. Alternativ können auch integrierte Pegelwandler zum Einsatz kommen.

Softwareseitig sind für beide Modulvarianten die folgenden Hinweise zu beachten:

Vor dem Senden mittels rf12_sendStart müssen eventuelle Empfangsvorgänge abgeschlossen werden. Dies erfolgt durch den Aufruf von

rf12_recvDone

Die Sendebereitschaft kann über

rf12_canSend

geprüft werden. Wird der Wert „1" zurückgeliefert, kann direkt danach mit

rf12_sendStart

die Übertragung gestartet werden.

Das zweite Argument von rf12_sendStart ist ein Zeiger auf die zu übertragenden Daten. Wenn ein Array versendet werden soll, muss hier lediglich der Array-Name eingesetzt werden. Bei einer einzelnen Variable muss noch das „&" vorgestellt werden. Das dritte Argument ist die Länge der Daten in Byte. Diese ist auf maximal 66 Byte begrenzt.

Zum Empfangen wird die Funktion rf12_recvDone aufgerufen. Um einen Datenverlust zu vermeiden, sollte dieser Aufruf regelmäßig und in geringen Zeitabständen erfolgen. Wird ein Wert ungleich Null zurückgegeben, dann wurde ein Datenpaket empfangen. Dieses wird in das Array rf12_data geschrieben. Auch hier sollte auf ein rasches Auslesen Wert gelegt werden, da die Daten beim Empfang des nächsten Pakets überschrieben werden. Damit ergibt sich der folgende Sketch für die Empfängerseite:

```
#include <JeeLib.h>
int counter;

void setup()
{ pinMode(8, OUTPUT);
  Serial.begin(9600);
  Serial.println("Rx-Setup");
  // node 10, group 212, 433 MHz
  rf12_initialize(10, RF12_433MHZ, 212);
}

void loop()
{ if (rf12_recvDone() && rf12_crc == 0)
  { Serial.print("Data received: ");
    Serial.print(rf12_data[0]+256*rf12_data[1]);
    Serial.println();
  }
}
```

Falls es zu Instabilitäten kommt, sollten die Anschlüsse nRES und nIRQ mit 10-kΩ-Widerständen an P5V gelegt werden.

nRF24L01-Module

Die nRF24L01-Module arbeiten bereits im Gigahertz-Band, die Sende-/Empfangsfrequenzen liegen bei 2,4 GHz. Dies bedeutet, dass sich die Wellenausbreitung immer mehr an die Eigenschaften von sichtbarem Licht angleichen. In diesem Frequenzbereich ist also eine direkte Sichtlinie zwischen Sender und Empfänger vorteilhaft. Betonwände oder Ziegelmauern werden wesentlich schlechter durchdrungen als bei niederen Frequenzen. Die Module eignen sich daher am besten beispielsweise für Roboterfernsteuerungen, bei welchen ohnehin eine Sichtverbindung zur steuernden Person vorhanden sein sollte.

In einer ersten Anwendung wird zunächst lediglich eine einfache „Hello World"-Nachricht von einem Arduino zum anderen gesendet. In einem zweiten Beispiel soll dann eine bidirektionale Kommunikation zwischen den Arduino-Boards aufgebaut werden. Dazu wird ein Joystick-Modul am ersten Arduino angeschlossen. Die Signale dieses Moduls werden über Funk zu einem anderen Arduino übertragen. Dort sollen sie einen Servomotor ansteuern. Am zweiten Arduino ist neben dem Servo auch ein Taster angeschlossen, der, wiederum über Funk, eine LED am ersten Arduino steuert.

Zunächst soll das nRF24L01-Transceiver-Modul etwas genauer betrachtet werden. Es arbeitet mit Baudraten von bis zu 2 Mbps. Bei Verwendung im freien Feld und mit geringer Baudrate kann die Reichweite bis zu 100 m betragen. Das Modul kann 125 verschiedene Kanäle verwenden. Dadurch wird der Betrieb eines Netzwerks mit 125 Sende-/Empfangsstationen an einem Ort ermöglicht. Jeder Kanal kann bis zu sechs Adressen verarbeiten. So kann jede Einheit mit bis zu sechs anderen Einheiten gleichzeitig kommunizieren. Die Leistungsaufnahme der Module beträgt während der Übertragung nur ca. 12 mA. Es wird also weniger Strom benötigt als für eine Standard-LED. Die Betriebsspannung des Moduls beträgt 1,9 bis 3,6 V. Die I/O-Pins sind jedoch 5-V-tolerant, sodass sie direkt an einen Arduino angeschlossen werden können, ohne dass Level-Shifter erforderlich sind. Die Kommunikation erfolgt wieder über den bekannten SPI-Bus. Der Interrupt-Pin wird in den folgenden Beispielen nicht verwendet. Weitere Details zum SPI-Bus finden sich in den letzten Artikeln zu dieser Serie. Die Tabelle 1 fasst alle notwendigen Verbindungen zusammen. Bild 5 zeigt wie das nRF24L01-Modul mit einem Arduino NANO verbunden wird.

Sobald die nRF24L01-Module an die Arduino-Boards angeschlossen sind, können die Sketches für den Sender und den Empfänger geladen werden. Hierzu ist die RF24-Bibliothek erforderlich, die unter

https://github.com/nRF24/RF24/tree/master/examples

aus dem Internet geladen werden kann.

Durch die Bibliothek wird die Programmierung wie üblich wesentlich vereinfacht. Das Programm für den Sender sieht damit so aus:

```
// nRF24L01_hello_world_TX.ino

#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>

// MOSI -> D11
// MISO -> D12
// SCK  -> D13
RF24 radio(8, 7); // CE, CSN

const byte address[6] = "00001";

void setup()
{ radio.begin();
  radio.openWritingPipe(address);
  radio.setPALevel(RF24_PA_LOW);
  radio.setRetries (1, 15);
  radio.stopListening();
}

void loop()
{ const char text[] = "Hello ELV";
  radio.write(&text, sizeof(text));
 }
```

Tabelle 1

nRF24L01 PIN	Signal	Arduino NANO
1	GND	GND
2	3V3	3V3
3	CE	D08
4	CSN	D07
5	SCK	D13
6	MOSI	D11
7	MISO	D12
8	IRQ	–

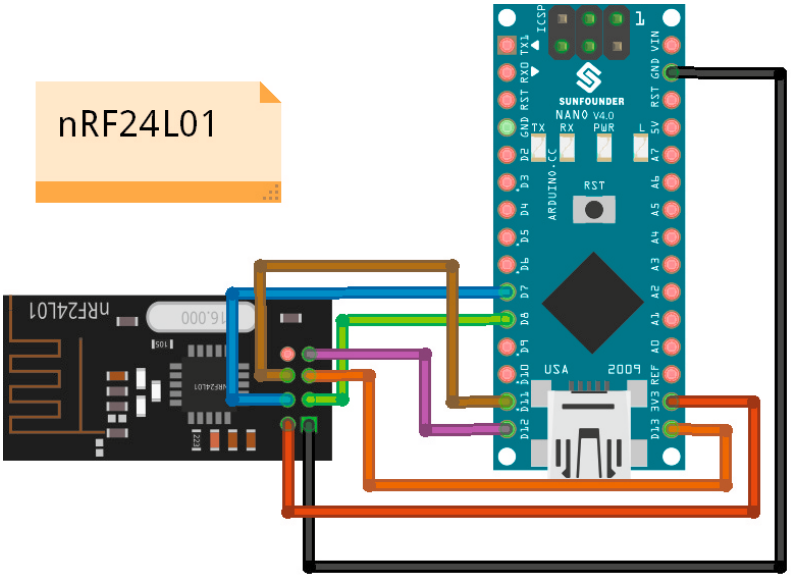

Bild 5: nRF24L01-Modul am Arduino NANO

Der Empfänger-Arduino ist mit dem folgenden Sketch zu laden:

```
// NRF24L01_hello_world_RX.ino

#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>

// MOSI -> D11
// MISO -> D12
// SCK  -> D13
RF24 radio(8, 7); // CE, CSN

const byte address[6] = "00001";

void setup()
{ Serial.begin(9600);
  Serial.println("RX starting up...");
  delay(1000);
  radio.begin();
  radio.openReadingPipe(0, address);
  radio.setPALevel(RF24_PA_LOW);
  radio.startListening();
}

void loop()
{ if (radio.available())
  { char text[32] = "";
    radio.read(&text, sizeof(text));
    Serial.println(text);
  }
}
```

Nach dem Einbinden der RF24-Bibliotheken wird jeweils ein RF24-Objekt erstellt. Die beiden Argumente für diese Objekte sind hier die CSN- und CE-Pins:

RF24 radio(8, 7); // CE, CSN

Als nächstes wird ein Byte-Array erstellt, das die eindeutige Übertragungsadresse enthält. Diese Adresse wird auch als „Pipe" bezeichnet und dient als virtueller Übertragungskanal für die beiden Module:

const byte address [6]="00001";

Die Adresse kann aus einem beliebigen 5-Zeichen-String bestehen. Sie ermöglicht es, einen bestimmten Empfänger anzusprechen. Wenn nur ein Sender-/Empfänger-Paar verwendet wird, muss auf beiden Seiten die gleiche Adresse eingegeben werden.

Im Set-up werden die Radio-Objekte initialisiert. Mit der Funktion radio.openWritingPipe() erfolgt die Adresseinstellung des Empfängers:

radio.openWritingPipe(address);

Auf der Empfängerseite kommt die gleiche Adresse zum Einsatz und sorgt für den Kommunikationsaufbau zwischen den beiden Modulen.

Mit der Funktion radio.setPALevel() wird die Ausgangsleistung der Senderendstufe eingestellt. Hier sind die folgenden Varianten verfügbar:

- radio.setPALevel(RF24_PA_MIN);
- radio.setPALevel(RF24_PA_LOW);
- radio.setPALevel(RF24_PA_HIGH);
- radio.setPALevel(RF24_PA_MAX);

Bei Verwendung höherer Ausgangsleistungen sollte ein Abblock-Kondensator in die Versorgungsspannung der Module eingeschleift werden. Dadurch werden Reichweitenprobleme meist deutlich entschärft. Die Funktion

radio.stopListening()

versetzt ein Modul in den Sendebetrieb. Das Gegenstück ist die Funktion

radio.startListening()

Sie definiert das entsprechende Modul als Empfänger.

In der Main-Loop wird ein Array aus Zeichen definiert, welche die zu sendende Nachricht enthalten. Mit

radio.write()

wird diese Nachricht an den Empfänger übertragen.

Mit der Funktion sizeof() wird die Länge des zu übertragenen Textes übergeben. Auf Empfänger-Seite, wird mit der Funktion

radio.available()

geprüft, ob Daten empfangen wurden. Wenn dies der Fall ist, können diese mit der Funktion

radio.read ()

gelesen werden. Schließlich wird der empfangene Text zur Überprüfung einer erfolgreichen Übertagung auf dem seriellen Monitor ausgegeben.

Bidirektionale Kommunikation

In einem zweiten Anwendungsbeispiel wird eine bidirektionale drahtlose Kommunikation zwischen zwei Arduino-Boards aufgebaut.

Anders als im vorherigen Beispiel werden nun zwei Pipes bzw. Adressen benötigt:

const byte address [] [6] = {"00001", "00002"};

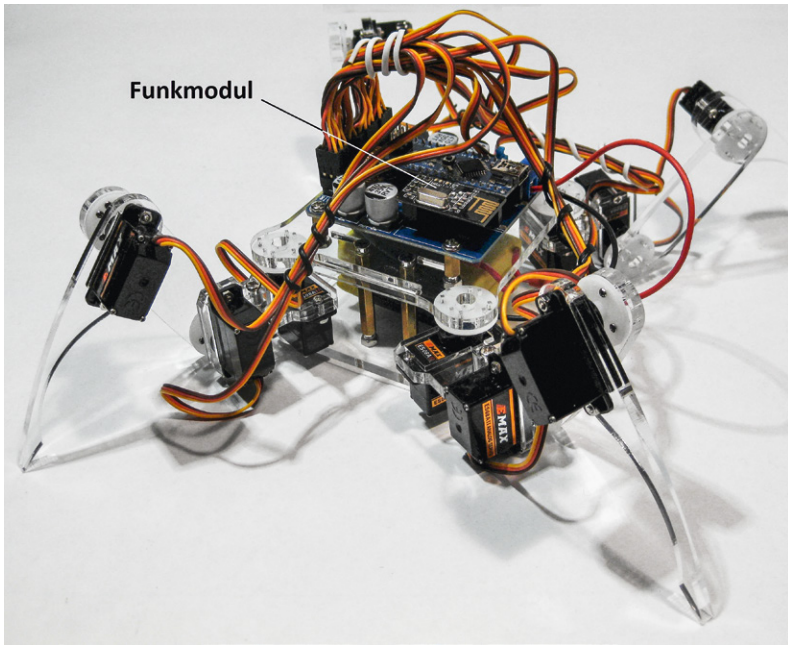

Bild 6: Quadruped mit nRF24L01-Fernsteuerung

Tabelle 2

	Serielle Funkmodule	RFM-Serie	nRF24L01-Module
Arbeitsfrequenz	433 MHz	433 oder 868 MHz	2,4-GHz-Band
Reichweite	Bis 30 m	Bis 100 m	Einige Meter
Softwareaufwand	Gering	Mittel	Mittel
Schnittstelle	Seriell	SPI	SPI

Weitere Infos:

- Grundlagen zur elektronischen Schaltungstechnik finden sich in der E-Book-Reihe „Elektronik!“ (www.amazon.de/dp/B000XNCB02)
- FRANZIS AVR-Mikrocontroller in C programmieren, Bestell-Nr. CR-09 73 52
- Elektor-Praxiskurs AVR-XMEGA-Mikrocontroller, Bestell-Nr. CR-12 07 62
- FRANZIS Physical Computing, Bestell-Nr. CR-12 21 81
- FRANZIS Lernpaket Motoren & Sensoren mit Arduino, Bestell-Nr. CR-12 74 74

Zu beachten ist, dass die Schreibadresse am ersten Arduino die Leseadresse beim zweiten Arduino ist und umgekehrt.
In der Main-Loop wird mit der Funktion
radio.stopListening()
der erste Arduino als Sender aktiviert.

Die Werte eines Joysticks werden mit der Funktion radio.write() an den Empfänger übertragen.

Auf der anderen Seite wird der Arduino mit der Funktion radio.startListening() als Empfänger gestartet. Wenn Daten empfangen wurden, werden diese an den Servo weitergeleitet.

Anschließend werden die Rollen von Sender und Empfänger getauscht. Nun wird das Signal eines Tasters zurück übertragen und steuert eine LED auf der anderen Seite der Funkstrecke.

Damit wurde demonstriert, dass die Datenübertragung zwischen den Arduinos in beide Richtungen funktioniert. Die Sketche zur bidirektionalen Datenkommunikation finden sich im Download-Paket zu diesem Artikel.

Dass man nicht nur einen Servo mit dem nRF24L01-Funkmodul übertragen kann, zeigt Bild 6. Hier dient das Modul zur Steuerung von zwölf Servos in einem Quadruped-Roboter.

Die Tabelle 2 liefert einen Überblick zu den in diesem Artikel vorgestellten Funkmodulen. Die Auswahl der passenden Funkstrecke für ein beliebiges Selbstbauprojekt sollte damit kein Problem mehr sein.

Ausblick

In diesem Beitrag wurden die Grundlagen der Funkübertragung von Mess- und Steuersignalen ausführlich dargelegt. Mehrere Varianten erlauben die optimale Auswahl eines Funksystems für jede vorgegebene Aufgabe.

Im nächsten Artikel geht es nochmals um Funktechnik, dann aber in einer etwas anderen Form. Mit sogenannten RFID-Modulen wird es möglich, mit einem Arduino RFID-Tags auszulesen. Diese Tags sind unter anderem als Karten oder als Schlüsselanhänger erhältlich. Damit lassen sich dann die verschiedensten Sicherheitsaufgaben lösen.

Neben den Grundlagen dieser Technologie werden wie immer auch verschiedene Praxisanwendungen vorgestellt. So entsteht etwa eine berührungslose Zugangskontrolle oder aber auch eine Lock-Box, die nur mittels einer speziellen Code-Karte geöffnet werden kann. ELV

Preisstellung Dezember 2017 – aktuelle Preise im ELV Shop

Empfohlene Produkte	Best.-Nr.	Preis
Arduino UNO	CR-10 29 70	€ 27,95
Velleman HF-Empfangsmodul, 5 V, -108 dBm, 433,92 MHz	CR-12 90 57	€ 8,95
Velleman HF-Sendemodul TX433N, 433,92 MHz	CR-12 90 58	€ 6,95

Alle Arduino-Produkte wie Mikrocontroller-Platinen, Shields, Fachbücher und Zubehör finden Sie unter: www.arduino.elv.de

Download-Paket zum Artikel:

Die Sketche und Beispieldateien zu diesem Artikel können kostenlos heruntergeladen werden unter www.elv.de: Webcode #10165