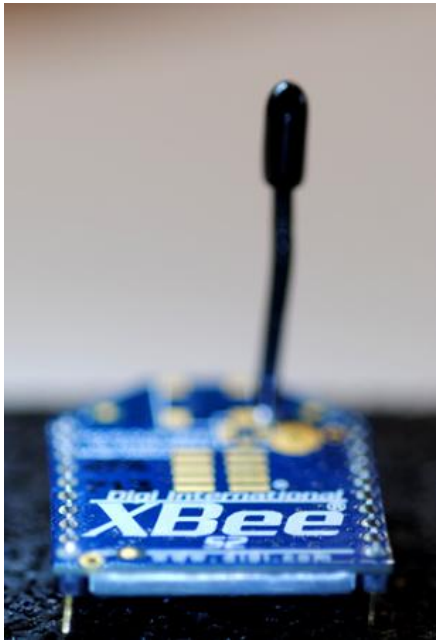
XBee Funkmodule



Daten kabellos übertragen

Markus Ulsaß, attraktor Hamburg, 4.3.2013

Agenda

- Was ist XBee?
- Anwendungsbeispiele mit XBee
- Vergleich Series 1/ Series 2
- Anschluss und Konfiguration
- Topologie und Betriebsmodi
- API-Modus
- Anwendungsbeispiele API-Modus
- Buchtipps und Links

XBee Series 2

Was ist XBee?

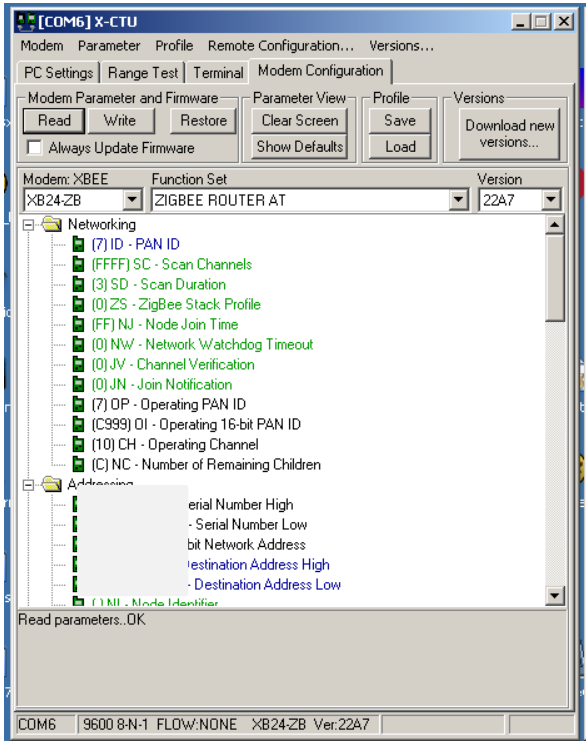
- **XBee** ist ein Markenname von Digi International für eine Familie von Funkmodulen mit einem bestimmten Bauteil-Format in Through-Hole und SMT-Technologie. Die Module sind größtenteils pinkompatibel.

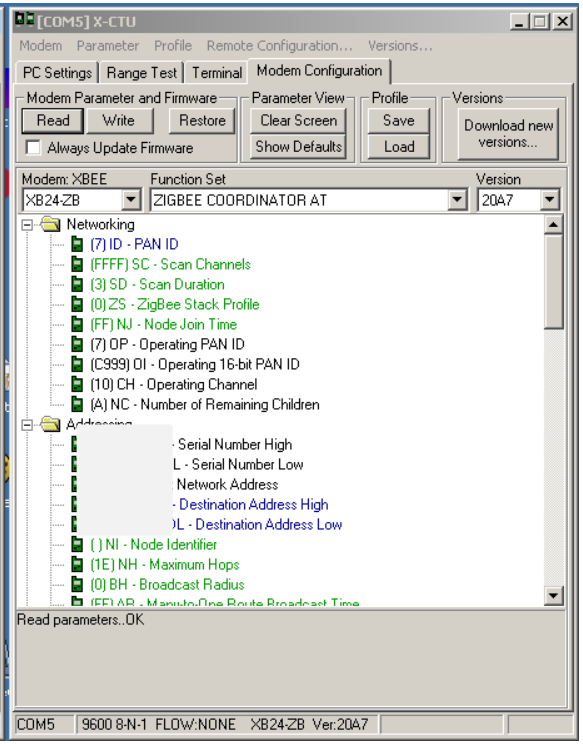
- Die Module eignen sich zum Übertragen von (seriellen) Daten über Funk - z.B. für Sensoren. Durch digitale In- und Output-Pins, sowie analoge Input-Pins kann u.U. auf einen Mikrocontroller zur Anbindung verzichtet werden (Bsp. autarker Temperatur-Sensor).

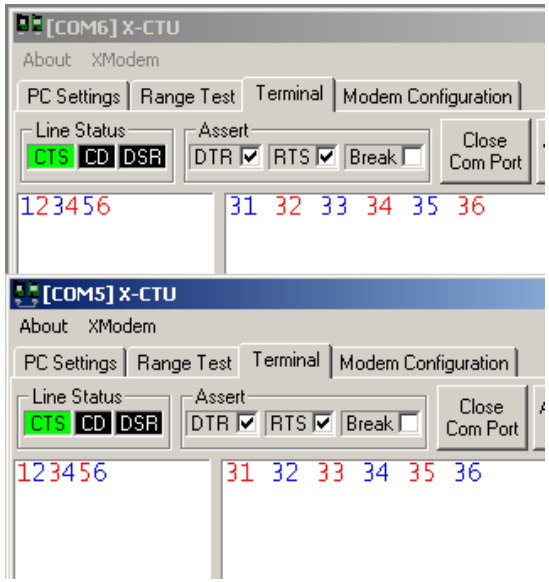
- XBee (ab Series 2) kann Zigbee (http://www.zigbee.org/) sprechen - ein Industriestandard (> 300 Unternehmen) für Funknetze, der es ermöglicht, kostengünstige, energiesparende Geräte in Netzwerken zu verbinden. Zigbee-Produkte gibt es seit 2005.

Anwendungsbeispiele

Einfache Übertragung im Transparent -(AT) Modus mit X-CTU







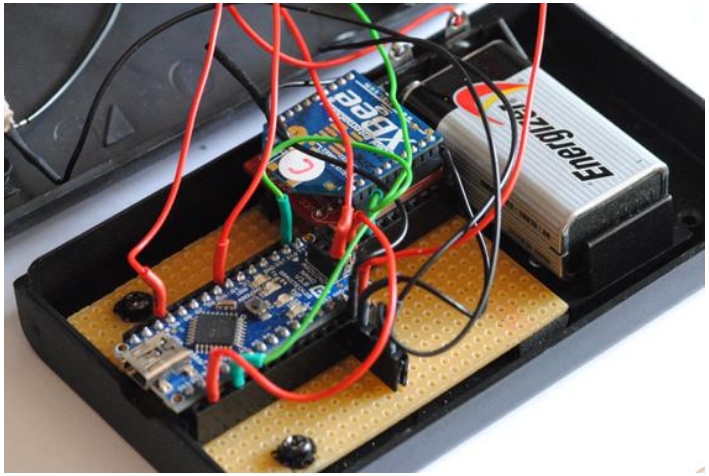
Blau = Input, Rot = Output

Anwendungsbeispiele

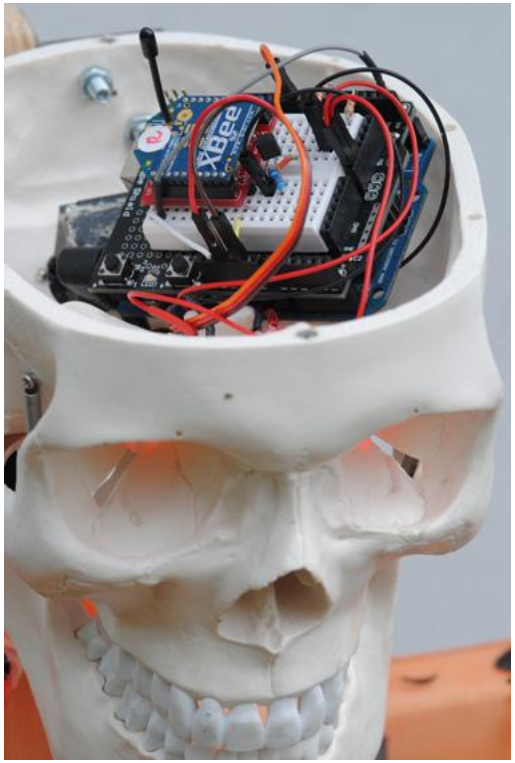
- XSkull (Halloween) http://www.youtube.com/watch?v=_YuO6nUENFo

Anwendungsbeispiele

XSkull



Sender (Arduino Nano, XBee S2)



Empfänger, (Arduino UNO, XBee S2)

XBee-Module Series 1

(Freescale Chipsatz)

- XBee Series 1 benutzt Protokoll 802.15.4 -> beschreibt ein Übertragungsprotokoll für Wireless Personal Area Networks (WPAN)):
- Point-to-Point/ Stern-/ proprietäres Mesh-Netzwerk von Digi, höhere Übertragungsrate (ca. 80kbps) und schneller als Series 2.

Datenblatt Digi: *The 802.15.4 XBee is significantly faster than ZigBee; RF latency can generally be calculated in 802.15.4. Throughput is also much higher, a practical maximum throughput is around 80kbps (Series 2: 35kbps)*

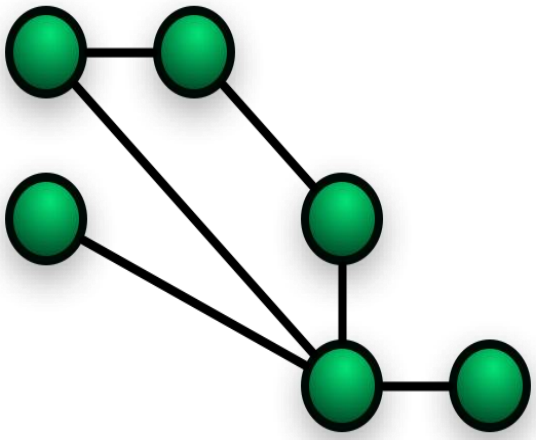
- Relativ einfach zu benutzen (Verbindung „out-of-the-box“)
- NICHT Zigbee-kompatibel
- NICHT kompatibel mit Series 2 Modulen (anderer Chipsatz/ Firmware)
- XBee-PRO: US- und internationale Version (Unterschied in der Sendeleistung)

XBee Series 2/ 2B (optimiert)/ PRO

(Ember Chipsatz)

Point-to-Multipoint/ Cluster-Tree (partial mesh)/ Zigbee Mesh Network:

Selbst-formendes, selbst-korrigierendes, dynamisches und mit verlässlichem Routing funktionierendes Netzwerk.
Problematisch bei: vielen Nodes oder unbekannten Routen und Adressen, wegen: Broadcast, Network Address Discovery, Route Discovery, wiederholtem Senden.
Langsamer als XBee Series 1 (max. 35kbps)



Mesh-Network, Quelle Wikipedia

-> Detaillierte Performance-Werte in kbps s. S. 64 XBee Product Manual

Merkmale (ZB - Series 2)

- Frequenz 2.4 GHz ISM-Band
- Jede XBee hat eine eindeutige 64-bit Geräteadresse
- Versorgungsspannung: 2,1 -3,6V bei ZB; „PRO“ 2,7-3,6V
- 11 **digitale** In-/ Outputs (auch als Interrupt), davon 4 als **analoge** Inputs nutzbar (10 bit Auflösung, max. sample rate 1kHz, Vref = 1,2V).
- Pins source und sink 4mA (außer: RSSI/PWM, DIO10, DIO4 digital Outputs 8mA); Iges max. 40mA
- Power-down im Sleep-Modus (Pin-/ Cyclic-Sleep) etwa 1µA (S1: 10 µA)
- Sicherheit: verschlüsselbar (128-bit AES encryption)
- 250 kbps Brutto-Übertragungsrate RF-Interface (netto deutlich geringer)
- Bidirektionale Kommunikation (im API-Modus)
- Pin pitch 2mm/ 0.1“ (evtl. Breakout-Board notwendig, ca. 2 Euro)
- Firmware über Funk programmierbar (API-Modus)

Antennentypen und weitere Module

- Wire/ Whip
- U.FL
- RPSMA
- PCB Antenne

Zahlreiche andere Produkte:

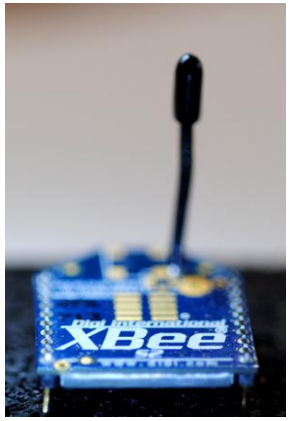
- 868/ 900MHz-Module, Reichweite bis zu 85 km (Line of sight)
- WiFi XBees
- Programmierbare XBees (Freescale MC9S08QE32- Prozessor, 32k Flash und 2k RAM mit Bootloader)
- Internet-Gateway-Produkte („Connect Port“) und Adapter

Anschluss und Konfiguration der Module

Verwendete XBee:

XB24-Z7WIT-04 Series 2

wire/ whip-Antenne (ca. 20 Euro)

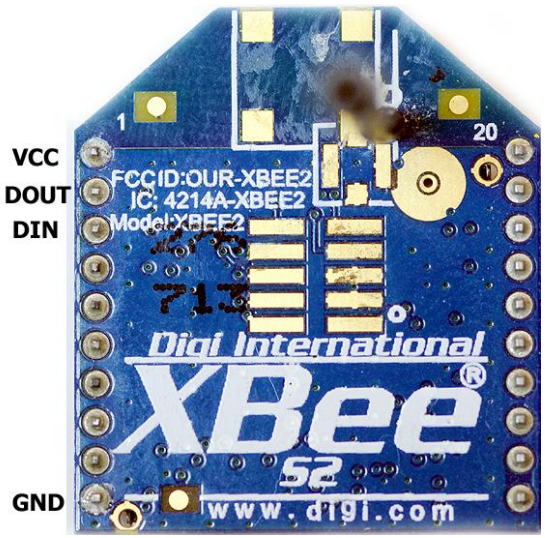


- Stromsparend (ca. 40mA Senden, < 1µA power down)
- Nicht so leistungsfähig (1mW => 40m (Indoor), 120 m (Outdoor)) wie die PRO mit 10mW (internationale Variante) => 90m/ 1500m (ca. 200mA)
- Großer Bereich der Versorgungsspannung (2,1- 3,6V,)

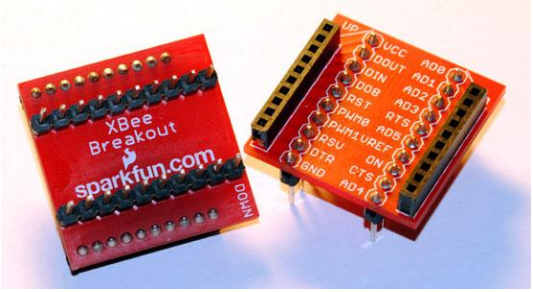
XBee-Pinout

Pin #	Name	Direction	Default State	Description
1	VCC	-	-	Power supply
2	DOUT	Output	Output	UART Data Out
3	DIN / $\overline{\text{CONFIG}}$	Input	Input	UART Data In
4	DIO12	Both	Disabled	Digital I/O 12
5	$\overline{\text{RESET}}$	Both	Open-Collector with pull-up	Module Reset (reset pulse must be at least 200 ns)
6	RSSI PWM / DIO10	Both	Output	RX Signal Strength Indicator / Digital IO
7	DIO11	Both	Input	Digital I/O 11
8	[reserved]	-	Disabled	Do not connect
9	$\overline{\text{DTR}}$ / SLEEP_RQ/ DIO8	Both	Input	Pin Sleep Control Line or Digital IO 8
10	GND	-	-	Ground
11	DIO4	Both	Disabled	Digital I/O 4
12	$\overline{\text{CTS}}$ / DIO7	Both	Output	Clear-to-Send Flow Control or Digital I/O 7. CTS, if enabled, is an output.
13	ON / $\overline{\text{SLEEP}}$	Output	Output	Module Status Indicator or Digital I/O 9
14	VREF	Input	-	Not used for EM250. Used for programmable secondary processor. For compatibility with other XBEE modules, we recommend connecting this pin voltage reference if Analog sampling is desired. Otherwise, connect to GND.
15	Associate / DIO5	Both	Output	Associated Indicator, Digital I/O 5
16	$\overline{\text{RTS}}$ / DIO6	Both	Input	Request-to-Send Flow Control, Digital I/O 6. RTS, if enabled, is an input.
17	AD3 / DIO3	Both	Disabled	Analog Input 3 or Digital I/O 3
18	AD2 / DIO2	Both	Disabled	Analog Input 2 or Digital I/O 2
19	AD1 / DIO1	Both	Disabled	Analog Input 1 or Digital I/O 1
20	AD0 / DIO0 / Commissioning Button	Both	Disabled	Analog Input 0, Digital IO 0, or Commissioning Button

XBee-Pinout, Quelle: Digi, XBee-Datenblatt, Rev. M



XBee Series 2



XBee Breakout Boards

USB-Adapter und Shields

SparkFun XBeeExplorer mit USB <-> Seriell (z.B. FTDI) (etwa 20 Euro)

Adafruit XBee Adapter ohne USB <-> Seriell (etwa 10 Euro)

XBee-Shields, Level-Shifter-Boards/ ICs

Arduino-Adapter Hack

Leerer Sketch (kein Bootloader)

void setup() { }

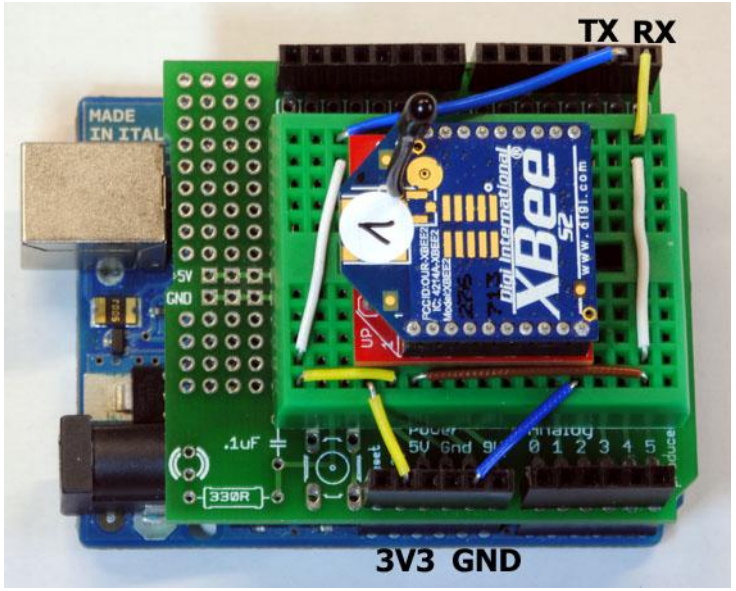
void loop() {}

VCC (XBee Pin 0) an 3,3V (Arduino)

GND (XBee Pin 10) an GND (Arduino)

DOUT (XBee Pin 1) an TX (Arduino Pin 1)

DIN (XBee Pin2) an RX (Arduino Pin 0)



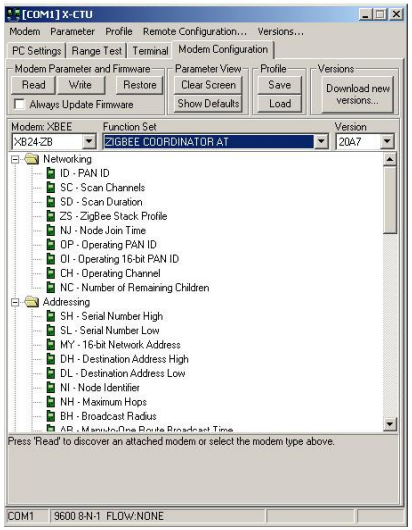
Arduino UNO XBee-Hack

Konfiguration

Digi XBee-Konfigurationstool X-CTU (WINDOWS ONLY)

- Konfiguration (Profile abspeichern, laden)
- Firmware-Updates
- Terminal-Funktion
- Range Test
- Remote-Konfiguration

User-Guide für X-CTU erhältlich.



X-CTU

Terminal Software (Win, Linux, Mac) - PuTTY, Tera Term etc.

Mit Terminal-Software ist aber z.B. kein Firmware-Update möglich, sondern nur reiner Kommando/ Transparent-Modus.

Topologie

Die XBees können auf drei verschiedene Arten konfiguriert werden:

Coordinator **Router** **End-Device**

Für ein Minimal-Netzwerk werden mindestens zwei Module benötigt. Eins davon **muss** als Coordinator konfiguriert werden – das andere als Router oder End-Device. Es gibt nur einen Coordinator im zugehörigen Netzwerk.

Der **Coordinator** ist für die Auswahl des Funkkanals, die PAN IDs (16- und 64-bit), die Sicherheitseinstellungen und das Netzwerkprofil verantwortlich.

Router können Daten senden, empfangen und weiterleiten. **End-Devices** können nur Daten senden und empfangen und sind besonders für Batteriebtrieb geeignet, da sie nicht ständig im Netzwerk vorhanden sein müssen.

Es sind Point-to-Multipoint, Cluster Tree (partial mesh) und Mesh-Netze (Zigbee) möglich.

Betriebsmodi

Command-Modus:

AT-Kommandos direkt an die XBee, z.B. zur Konfiguration.

Transparent-Modus (AT-Modus):

XBee schleift Datensignale durch – als Ersatz für eine Kabelverbindung.

API-Modus:

Alternativ zum Transparent-Modus ist der API-Betrieb (Application Programming Interface) möglich. Hierbei muss die Kommunikation strukturiert erfolgen, d.h. die Daten werden in bestimmten, vordefinierten Datenpaketen übertragen. Der API Modus spezifiziert wie Kommandos, Antworten darauf und Statusnachrichten versendet und empfangen werden. Vorteil: kein langsamer Command-Modus notwendig, bidirektional, Datenpakete können weitergeleitet und bestätigt werden, es können direkt Module angesprochen werden, I/O Samples können versendet werden.

API Frame-Namen und -Werte

AT Command	**0x08**
AT Command - Queue Parameter Value	0x09
ZigBee Transmit Request	0x10
Explicit Addressing ZigBee Command Frame	0x11
Remote Command Request	0x17
Create Source Route	0x21
AT Command Response	0x88
Modem Status	0x8A
ZigBee Transmit Status	0x8B
ZigBee Receive Packet (AO=0)	0x90
ZigBee Explicit Rx Indicator (AO=1)	0x91
ZigBee IO Data Sample Rx Indicator	0x92
XBee Sensor Read Indicator (AO=0)	0x94
Node Identification Indicator (AO=0)	0x95
Remote Command Response	0x97
Over-the-Air Firmware Update Status	0xA0
Route Record Indicator	0xA1
Many-to-One Route Request Indicator	0xA3

API-Modus Beispiel

AT command 0x08 (immediate on local radio)

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x04	
	Frame-specific Data	Frame Type	3	0x08	
		Frame ID	4	0x52 (R)	Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to 0, no response is sent.
		AT Command	5	0x4E (N)	Command Name - Two ASCII characters that identify the AT Command.
			6	0x4A (J)	
		Parameter Value (optional)			If present, indicates the requested parameter value to set the given register. If no characters present, register is queried.
	Checksum		7	0x0D	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

XBee Datenblatt Rev. M

Datenpaket: 0x7E 0x00 0x04 0x08 0x52 0x4E 0x4A 0x0D

Datenpakete mit Frames sind byteorientiert (hexadezimale Notation).

AT und API-Modus können gemischt in einem Netzwerk verwendet werden.

Übertragung im API-Modus

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x14	
	Frame-specific Data	Frame Type	3	0x92	
		64-bit Source Address	MSB 4	0x00	64-bit address of sender
			5	0x13	
			6	0xA2	
			7	0x00	
			8	0x40	
			9	0x52	
			10	0x2B	
			LSB 11	0xAA	
		16-bit Source Network Address	MSB 12	0x7D	16-bit address of sender.
			LSB 13	0x84	
		Receive Options	14	0x01	0x01 - Packet Acknowledged 0x02 - Packet was a broadcast packet
		Number of Samples	15	0x01	Number of sample sets included in the payload. (Always set to 1)
		Digital Channel Mask*	16	0x00	Bitmask field that indicates which digital IO lines on the remote have sampling enabled (if any).
			17	0x1C	
		Analog Channel Mask**	18	0x02	Bitmask field that indicates which analog IO lines on the remote have sampling enabled (if any).
		Digital Samples (if included)	19	0x00	If the sample set includes any digital IO lines (Digital Channel Mask > 0), these two bytes contain samples for all enabled digital IO lines. DIO lines that do not have sampling enabled return 0. Bits in these 2 bytes map the same as they do in the Digital Channels Mask field.
			20	0x14	
		Analog Sample	21	0x02	If the sample set includes any analog input lines (Analog Channel Mask > 0), each enabled analog input returns a 2-byte value indicating the A/D measurement of that input. Analog samples are ordered sequentially from AD0/DIO0 to AD3/DIO3, to the supply voltage.
			22	0x25	
	Checksum		23	0xF5	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

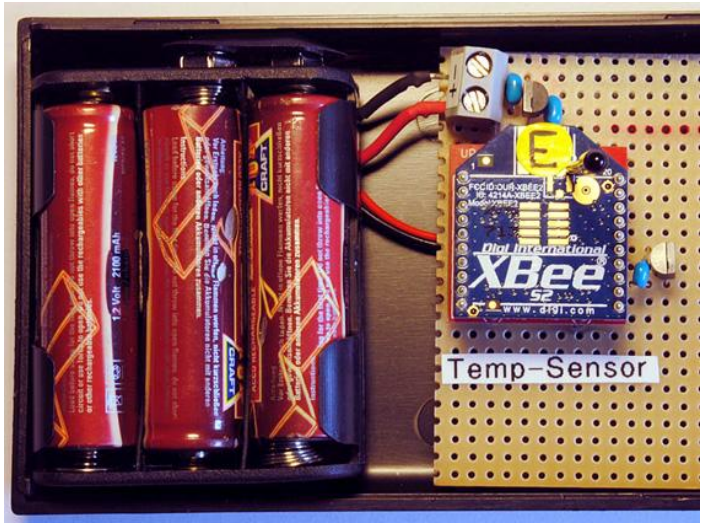
XBee Datenblatt Rev. M

Bsp. für die Übertragung eines I/O Data Samples von einem Sensor
(API Frame Type 0x92)

Digital-Pins D2 –D4 sind Input (digital channel mask 0x1C = B00011100) und Analog-Pin A1 ist Input (analog channel mask 0x02 = B0000010).
Digital Sample Mask : 0x00 (high byte), 0x14 = B00010100 (low byte) -> D2 und D4 sind „HIGH“
Analog Sample Mask: 0x02 (high) 0x25 (low) = 547

Anwendungsbeispiel API-Modus

Autarker Temperatursensor & Basisstation

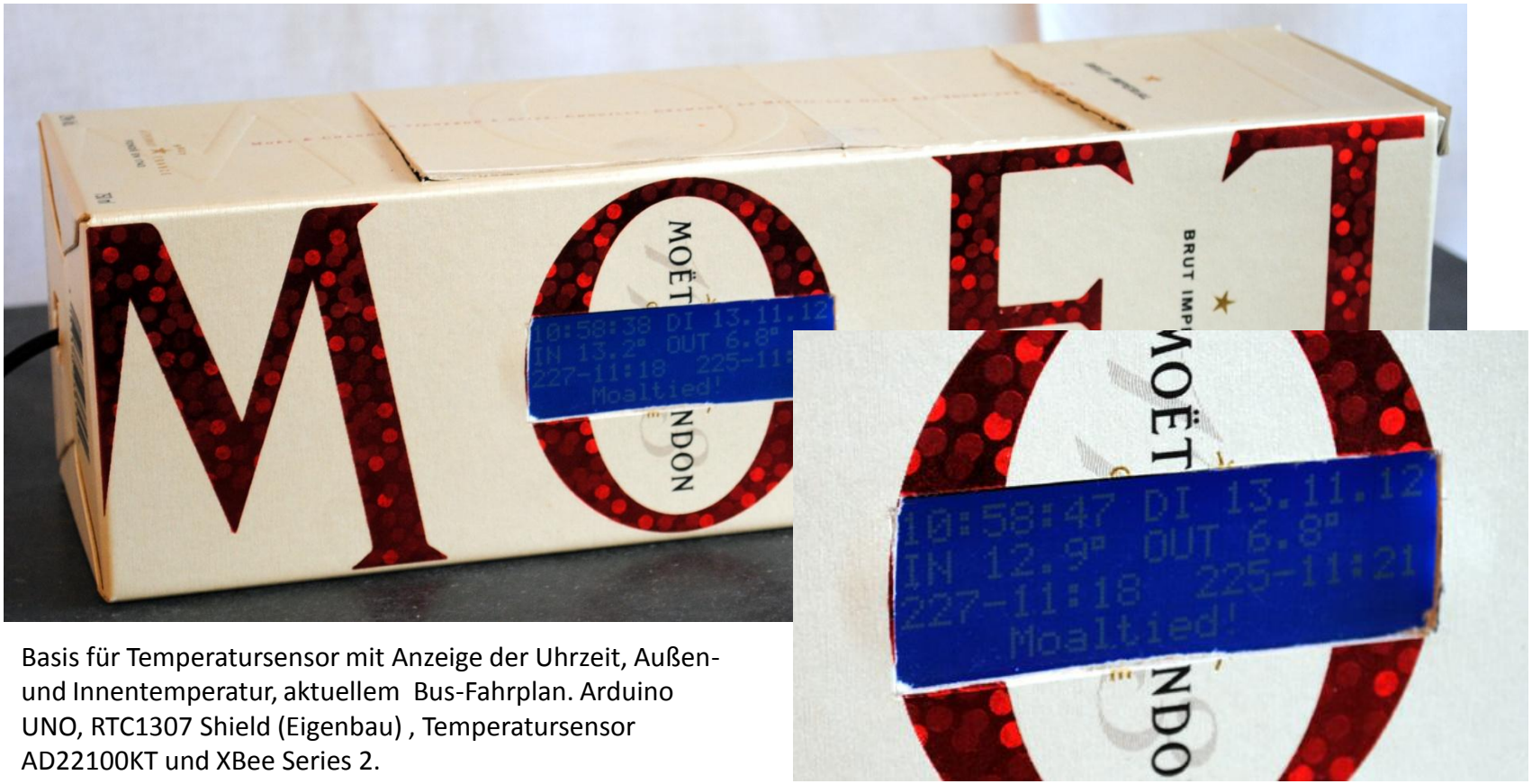


Temperatur-Sensor, XBee Series 2, MCP1700 (Low-Dropout Voltage Regulator 3,3V), Temperatursensor TMP36

Die XBee Series 2 ist als End-Device konfiguriert und wird im cyclic sleep-Modus betrieben. Das Modul sendet alle 4 Minuten ein I/O-Sample (API Frame 0x92) mit der aktuellen Temperatur, die durch den TMP36-Temperatursensor ermittelt wird. Spannungsversorgung über drei AA-Zellen und einen low-dropout Spannungsregler (MCP1700). Energieverbrauch im sleep-Modus: ca. 2,3 µA.

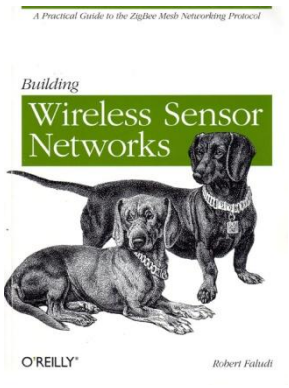
Anwendungsbeispiel API-Modus

Basisstation



Basis für Temperatursensor mit Anzeige der Uhrzeit, Außen- und Innentemperatur, aktuellem Bus-Fahrplan. Arduino UNO, RTC1307 Shield (Eigenbau) , Temperatursensor AD22100KT und XBee Series 2.
Design: WAF ;)

Bücher und Links



Buchtipp:

Robert Faludi, „Building Wireless Sensor Networks“,
O'Reilly, ISBN 978-0-596-80773-3, 2011

Links:

- XBee Product Guide *https://www.sparkfun.com/pages/xbee_guide*
- Zigbee-Standard *http://de.wikipedia.org/wiki/ZigBee*
- Product Manual XB Series
  http://ftp1.digi.com/support/documentation/90000976_M.pdf
- X-CTU
  http://www.digi.com/support/productdetail?pid=3352&type=utilities
- XBee Java API library *http://code.google.com/p/xbee-api/*
- XBee Arduino API library *http://code.google.com/p/xbee-arduino/*
- XBee-FAQ *http://www.jsjf.demon.co.uk/xbee/faq.pdf*

Blog:

Xbee Blog http://lookmanowire.blogspot.de