Roboter selbst bauen und programmieren –

Roboterbausatz NIBObee

Kleine Roboter selbst programmieren – das liegt im Trend, lassen sich doch so spielerisch Erfahrungen bei der Programmierung von Mikrocontrollern sammeln, die unmittelbar anwendbar sind. Nicht zuletzt das stark ausgeweitetete ELV-Angebot zu Roboter-Bausätzen und entsprechender Literatur zeugt von stark wachsender Beliebtheit des Themas.
Meist ist der Roboter dazu als Fahrroboter ausgeführt, diese Ausführung ist besonders einfach steuerbar und bietet natürlich sofort hervorragende Möglichkeiten für die Programmierung einer Steuerung.
nicai-systems bietet mit dem NIBObee einen besonders interessanten Roboterbausatz an – hier steht vor der ersten Fahrt der komplette Aufbau des Roboters.

Roboter praktisch

nicai-systems, eine kleine Firma aus Stolberg in der Nähe von Aachen, hat es sich vor einigen Jahren zum Ziel gesetzt, frei programmierbare, mobile Roboter zu entwickeln, die es vor allem jungen Leuten, also Schülern, Auszubildenden und Studierenden, einfach machen sollte, in die praktische Mikrocontrollerprogrammierung einzusteigen.

Ein Breadboard mit einer µC-Schaltung oder eine einfache Experimentierplatine wie etwa das myAVR-System, der sich als erstaunlich vielseitig nutzbar erweisende Ping-Pong-Bausatz von Franzis oder aber die Arduinos sind eine feine Sache – dennnoch hat etwas, was zum Schluss agiert, bei dem man Mess-, Steuer- und Regelungstechnik unmittelbar anwenden kann, eine hohe Faszination. Symptomatisch sind dabei immer wieder in der Vergangenheit Fahrroboter-Projekte wie der P!MOT oder der (noch aktuelle)

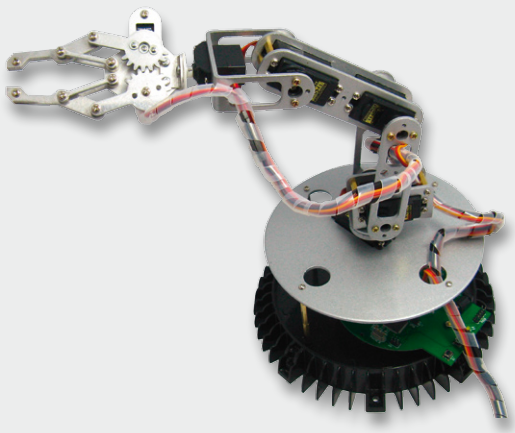

Bild 1: Vorbild: die Industrie. Mit solchen, der Realität nachgebildeten Industrierobotermodellen kann man komplexe Programmierabläufe für Produktionsroboter lernen. Quelle: AREXX

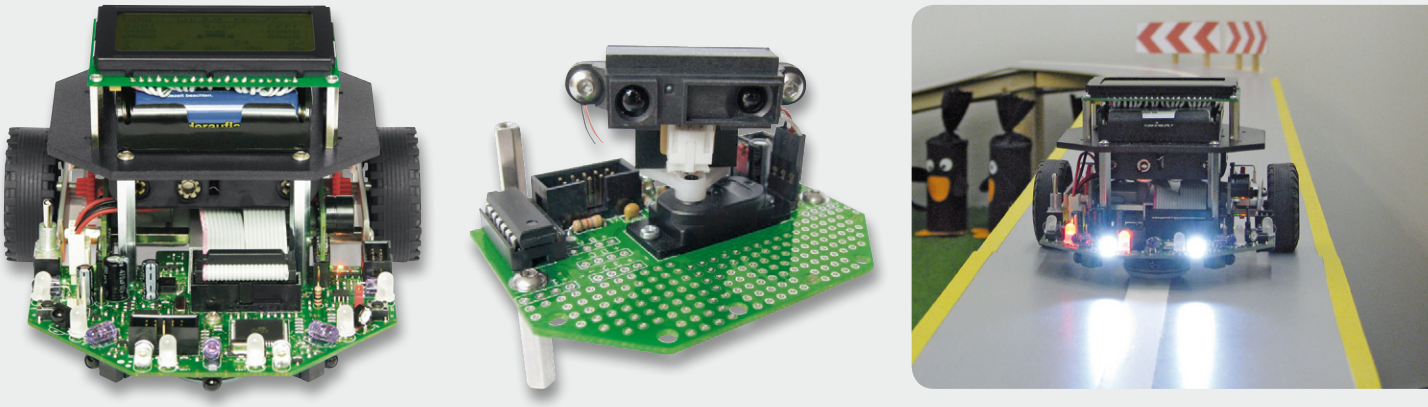

Bild 2: Der große Bruder des NIBObee, der NIBO 2, ein bereits sehr komplex aufgebauter Bausatz, der u. a. mit einem beweglichen Distanzscanner (Bildmitte) aufgerüstet werden kann und hervorragend als autonomer Fahrroboter agieren kann (Bild rechts). Quelle: nicai-systems

ASURO gewesen – man konnte eben sofort komplexe Steuer- und Regeltechnik live erleben, sehen, wie sich Programmierfehler auswirken, usw. Und diese Art des Roboters, der intelligent und autonom agiert, findet sich auch in zunehmender Zahl in der wirtschaftlichen Praxis – vom selbstfahrenden Containertransporter im Hafen bis zum selbst einparkenden Pkw, dem autonom dem Postboten folgenden Paketauto oder aber dem Service-Roboter in Krankenhaus und Altenpflege.

Auch die frei programmierbaren Industrieroboter, die es inzwischen auch schon als Lernmodell (Bild 1) [1] gibt, haben eine hohe Faszination für den, der sich mit Mikrocontroller-Programmierung beschäftigt.

Doch zurück zu nicai-systems. Der große Bruder des hier vorgestellten NIBObee, der NIBO 2 (Bild 2) [2], ist ein erfolgreiches Objekt für Ausbildung und Forschung. Er ist sehr großzügig mit Mikrorechentechnik ausgestattet und kann so sehr vielseitig eingesetzt werden. Mit zahlreichen Sensoren, u. a. einem motorisch stellbaren Distanzsensor, kann er sich frei im Raum orientieren und Fahraufgaben frei und intelligent erledigen – ohne mechanische Fühler und ohne anzustoßen, herunterzufallen etc. Zusammen mit dem optionalen Zubehör wie Programmieradapter, Grafikdisplay-Zusatz und dem (genialen) Distanzscanner gerät das Ganze aber für normale Jugendliche und Auszubildende schnell preislich aus dem Rahmen, hier sind wohl eher Studierende sowie Bildungseinrichtungen die Zielgruppe für dieses hochwertig mit SMD bestückte und robust ausgeführte Robotermodell.

Konsequent einsteigerfreundlich

Um auch die o. a. Zielgruppen mit einem technisch ähnlich gelagerten Modell bedienen zu können, entstand der kleine Bruder des NIBO 2, der NIBObee (Bild 3) [3]. Der trifft in dieser Beziehung den Nagel auf den Kopf: Für knapp 50 Euro erhält man einen wirklich kompletten Bausatz, der, bis auf die Akkus, alles enthält, was man braucht: ein von der Pike auf selbst aufzubauendes Modell, einsteigerfreundlich-konsequent in bedrahteter Technik ausgeführt, mit zwei großzügig dimensionierten Mikrocontrollern bestückt, dazu mit einer beachtlichen Grundausstattung an verschiedenen Sensoren, integriertem USB-Programmer, der auch gleich als Ladegerät für die Antriebsakkus dient. Um das Modell sofort in Betrieb nehmen zu können, ist bereits ein Testprogramm im Mikrocontroller enthalten, so kann man bereits vor dem ersten eigenen Programm sicher sein, dass die Hardware komplett funktioniert.

Als Hauptprozessor fungiert ein ATmega16 (Bild 4) mit 16 KB Flash-Speicher und 1 KB SRAM, der mit einer Taktfrequenz von 15 MHz betrieben wird. Als USB-Programmer/Akkulader ist ein ATtiny44 in das System implementiert.

Bereits in der Grundausstattung kann man sich mit zahlreichen Sensoren an Bord beschäftigen: 4 Tastsensoren mit zwei mechanischen Fühlern, 2 Odometriesensoren (für die Antriebssteuerung zuständig) sowie ein Liniensensor mit IR-LEDs und Phototransistoren bieten fürs Erste bereits ein weites Betätigungsfeld.

Über Erweiterungsports sind zahlreiche Erweiterungen wie Hinderniserkennungen, RC5-Fernsteuer-Erweiterung oder ein Grafikdisplay samt Bedientastern anschließbar. Darauf kommen wir noch im Einzelnen. Dazu kommt noch ein vorprogrammiertes Controller-Tuning-Kit, das noch mehr Speicher bietet.

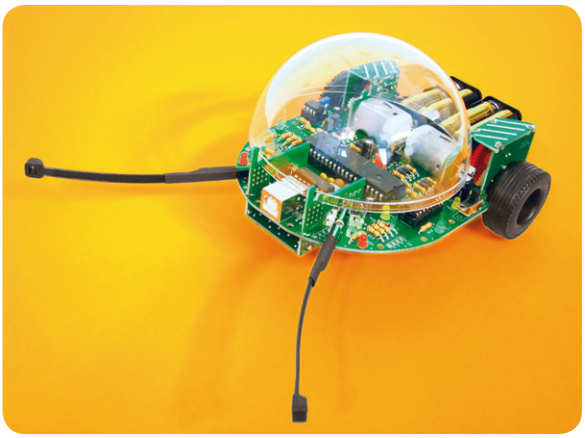

Bild 3: Der NIBObee, ein preiswerter, mit zwei AVR-Prozessoren und zahlreichen Sensoren ausgestatteter, erweiterbarer Fahrroboter-Bausatz. Quelle: nicai-systems

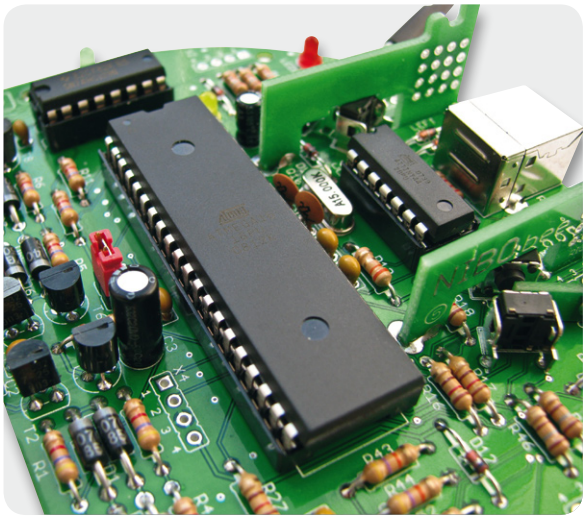

Bild 4: Das Herz des NIBObee: ein ATmega16 als Hauptprozessor und ein ATtiny44 als USB- und Ladecontroller. Quelle: nicai-systems

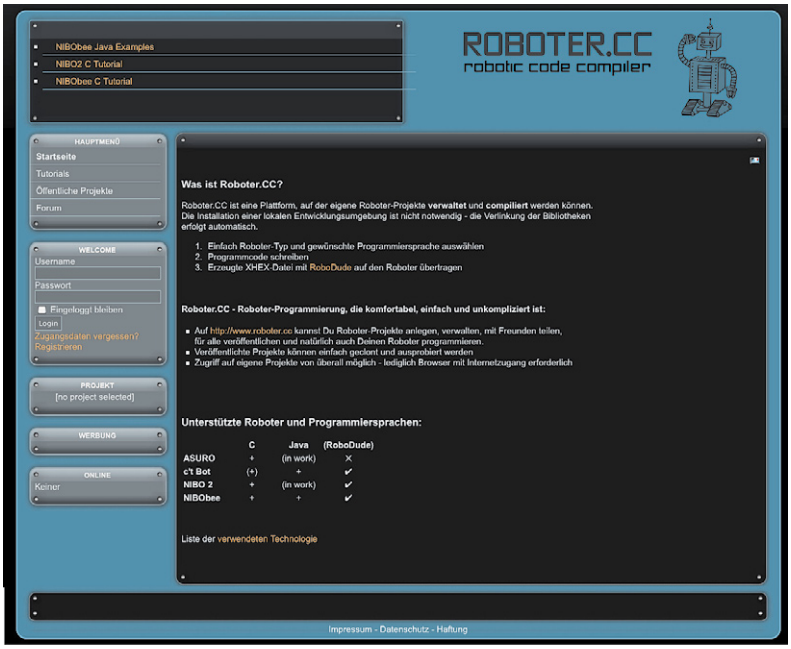

Bild 5: Mehr als nur ein Treffpunkt der Roboter-Fans – Roboter.CC bietet eine besonders bedienfreundliche IDE samt Compiler und Online-Archiv für Projekte.

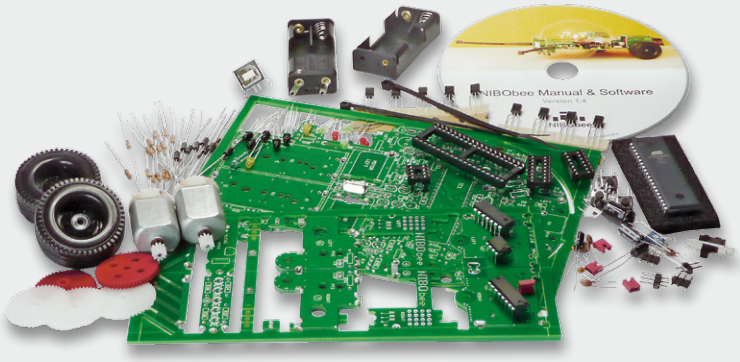

Bild 6: Vor dem Programmieren kommt das Löten – ein Teil des NIBObee-Bausatzes in der Übersicht.

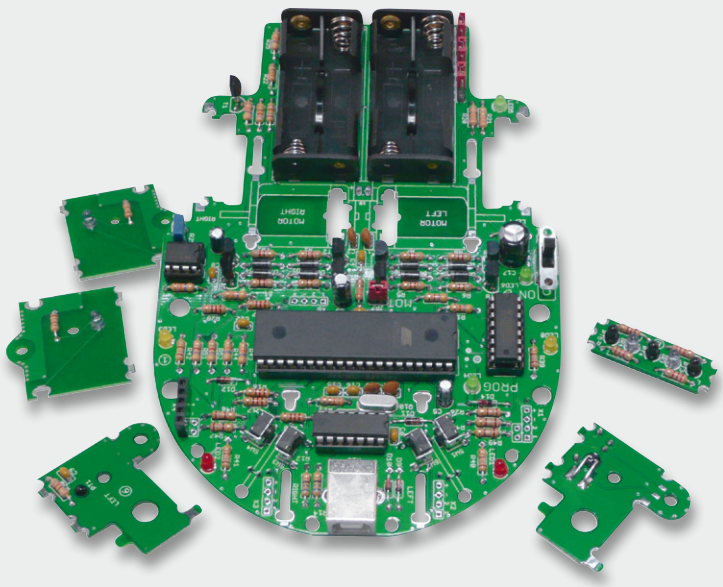

Bild 7: Alle Platinen sind bestückt, hier sind zur Demonstration auch schon die ICs aufgesteckt.

Die Programmierumgebung

Unterstützt wird das Projekt durch die Internetseite www.roboter.cc. Hier (Bild 5) kann man für verschiedene Roboter-Plattformen Programmcodes in unterschiedlichen Programmiersprachen (C/Java) erstellen, auf dem Server compilieren lassen, das Hex-File herunterladen und dann auf den Roboter übertragen. So kann man sich fürs Erste die Installation und das Erlernen einer eigenen Entwicklungsumgebung sparen. Zudem trifft man hier auf eine Community von Gleichgesinnten und findet zahlreiche fertige Software-Projekte sowie alle Anleitungen zu den jeweiligen Robotern, C-Tutorials usw.

Es gibt auch auf der mitgelieferten CD-ROM ein ausführliches Tutorial zum NIBObee, das Schritt für Schritt durch die Installation und die Arbeit mit der Atmel-Programmierumgebung AVR Studio und WinAVR (GCC, Libs und AVRDUDE) begleitet. Hierfür liefert nicai-systems eine NIBObeelib mit, die C/C++-Routinen für die Ansteuerung des Roboters, Treiber, Sensor-Kalibrierprogramme, Beispielprogramme und einen Programmer für die Datenübertragung auf den Roboter enthält.

Somit ist für maximale Unterstützung gerade des Einsteigers gesorgt.

Erster Schritt – Löten!

Auf keine andere Weise lernt man seinen Roboter besser kennen: selbst aufbauen!

Dafür kommt der NIBObee sauber in Einzelteilen, in unzähligen Tütchen verpackt, ins Haus. Bild 6 zeigt nur einen Teil der weit über 100 Bauteile.

Vorab sei gesagt: perfekt bis ins Detail! Es fehlt nichts. Die große Grundplatine kommt mit zahlreichen kleinen, abzubrechenden Platinen daher, diese dienen später als Sensor- und Zusatzbaugruppenträger sowie als Getriebegehäuse. Denn alle mechanischen Halterungen und Träger am NIBObee bestehen aus Platinenmaterial – der gesamte Roboter wird ohne eine Schraube zusammengebaut, nur gelötet – genial!

Eine der Abbrech-Platinchen bleibt übrig – nicht wegwerfen, sie ist mit Auskerbungen und Bohrungen für das Biegen der Bauteilanschlüsse und Transistorbeine versehen und dient so als Biegelehre.

Das Bestücken der Platinen macht Spaß. Nicht nur, weil alles da ist und sauber passt, auch die Anleitung ist hervorragend, gibt dazu dem Elektronik-Einsteiger viele Tipps, so kann man etwa nebenher die Identifizierung von Widerständen anhand des Farbcodes erlernen. In Bild 7 sieht man alle bestückten Platinen, jetzt ist der Roboter fertig zum Zusammenbau. Die kleine Platine rechts trägt den Liniensensor, die vier anderen die Odometriesensoren. Sie bilden gleichzeitig die Getriebegehäuse, wie wir noch sehen werden. Hier sind auch schon der Kasten für die Akkus und die ICs bestückt. Apropos Akkus: Es dürfen später im Betrieb keine 1,5-V-Batterien verwendet werden, die Schaltung ist für den Betrieb mit max. 5 V ausgelegt. Bild 8 zeigt einen Schaltungsausschnitt dazu, es befindet sich kein DC/DC-Wandler an Bord! Der Akkubetrieb hat auch den Vorteil geringerer Betriebskosten, die Akkus sind dazu komfortabel über den USB-Port wiederaufladbar.

Ist die Bestückung erfolgreich verlaufen, erfolgt das Einsetzen des Sensorträgers auf der Vorderseite sowie der Getriebe-Motor-Einheiten. Diese bestehen tatsächlich, neben den benötigten Getriebeteilen und den Motoren, allein jeweils aus drei Platinenteilen. Diese sind so geschickt gestaltet, dass man sie recht einfach und ohne große Mengen an ohnehin dann zu weichem Lötzinn zu einem Getriebegehäuse zusammenlöten kann. Das Verlöten erfolgt durch geschickte Nutzung der Kapillarwirkung, so dass die Teile hinterher quasi „verschweißt" sind. Deshalb muss man allerdings auch auf exakt senkrechten bzw. parallelen Stand aller Platinen zueinander achten, sonst klemmt das Getriebe und man erhält einen zumindest schwergängigen Motorlauf. Denn das Zerlegen des einmal verlöteten Getriebegehäuses gestaltet sich nicht ganz einfach.

Hier ist der im Vorteil, der über eine Heißluftstation mit feiner Heißluftdüse verfügt, um die Lötstellen gezielt und punktuell wieder lösen zu können, ohne Zahnräder etc. zu beschädigen. Exaktes Arbeiten und Kontrolle der Freigängigkeit des Getriebes während des Lötens zahlt sich also aus.

Nach dem Einbau der Fühler für die Tastsensoren und Einsetzen aller Schaltkreise in die Fassungen kann man dank der bereits auf dem AVR vorhandenen Testsoftware sofort einen ersten Funktionstest der Antriebe und des Liniensensors sowie der Tastsensoren anhand der LEDs auf dem Bord und der Reaktionen des Antriebs testen.

Verläuft dieser Test erfolgreich, ist nur noch die kleine Kunststoffhalbkugel auf der Unterseite der Platine anzukleben, Sie dient als „Vorderachse" des Fahrroboters. Bild 9 zeigt den so fertig aufgebauten Roboter. Dann kann es an das Programmieren, Testen und die ersten Fahrmanöver des NIBObee gehen!

Erste Programmierschritte mit Anleitung

Die Programmierung des NIBObee erfolgt in C/C++, aber auch das Programmieren via Java ist möglich. Wie bereits erwähnt, gibt es zwei Möglichkeiten hierzu. Die eine, nennen wir sie die klassische Variante, ist die über die originäre Atmel-Entwicklungsumgebung (IDE) AVR Studio in Zusammenarbeit mit WinAVR.

Über eine auf der mitgelieferten CD-ROM zu installierende eigenständige Programmer-Software, den „NIBObee Programmer", kann die fertige Hex-Datei via USB sehr einfach auf den Roboter übertragen werden. Man muss sich also erst einmal nicht mit ISP-Schnittstellen, Seriell-USB-Umsetzung u. Ä. beschäftigen.

Hat man neben der NIBObeelib (die übrigens in jedem Falle zu installieren ist, sie enthält u. a. den USB-Treiber des Roboters) den Programmer installiert, kann man sofort die mitgelieferten und fertig compilierten Programmbeispiele auf den Roboter übertragen (Bild 10) und so z. B. den Liniensensor kalibrieren. Wer bereits AVRDude auf seinem Rechner installiert hat, kann auch diese Programmer-Software am NIBObee nutzen.

Dann kann es an die ersten eigenen C-Projekte gehen. Schritt für Schritt führt das Tutorial den Einsteiger in das Anlegen eines neuen Projektes ein, bis

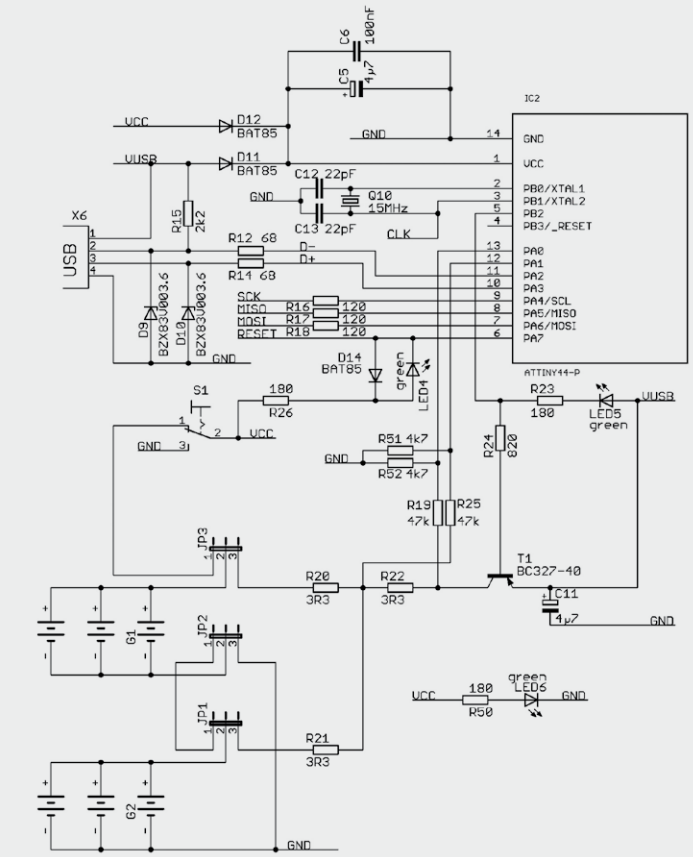

Bild 8: Der Schaltungsausschnitt aus der NIBObee-Schaltung zeigt die Spannungsversorgung und den USB-Anschluss samt USB- und Ladeprozessor.

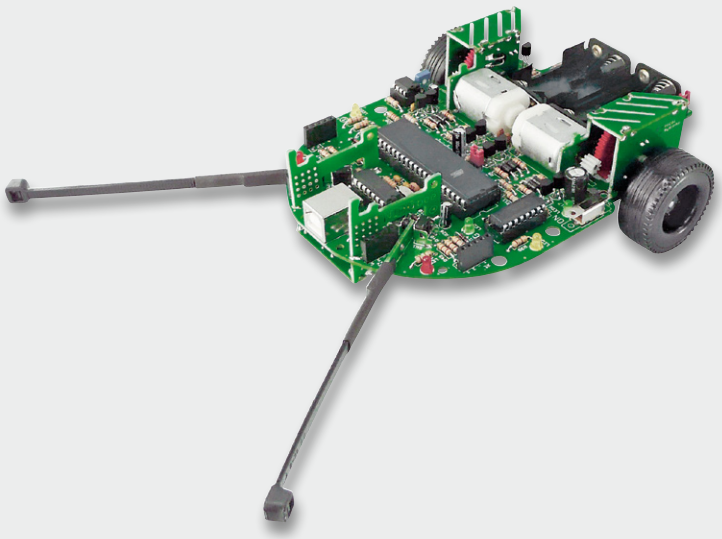

Bild 9: Unser fertig aufgebauter Roboterbausatz

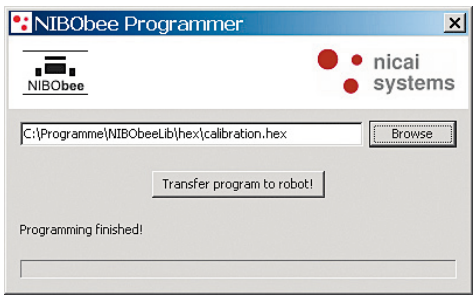

Bild 10: Mittels des NIBObee-Programmers werden die per AVR-Studio-IDE erzeugten Daten an den Roboter übertragen.

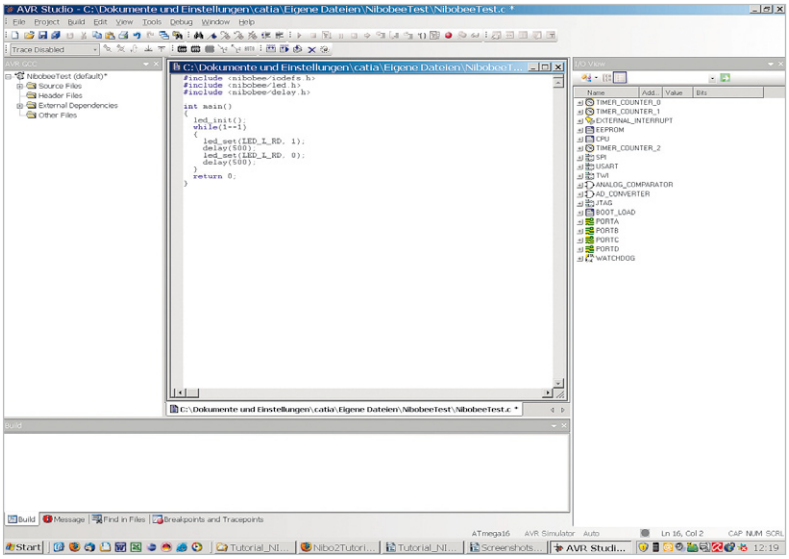

Bild 11: Die kostenlos bei Atmel erhältliche Standard-Programmierumgebung AVR Studio in Aktion mit einem NIBObee-Demoprogramm.

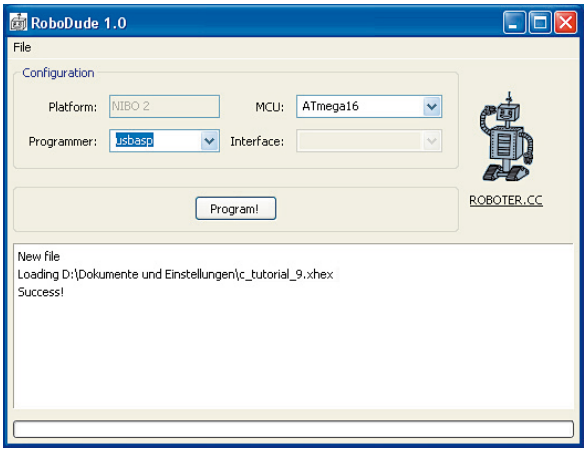

Bild 12: RoboDude sorgt für das Programmieren der Xhex-Pakete aus der Roboter.CC-Plattform auf den NIBObee.

schließlich der erste Code in der IDE steht (Bild 11). Ist der compiliert, wird er mittels der Programmer-Software auf den Roboter übertragen. Übrigens lohnt es sich, ab und an in die Download-Sektion von nicai-systems zu sehen, auf der Open-Source-Seite „sourceforge.net" gibt es von Zeit zu Zeit eine neue NIBObeelib.

Das Tutorial kann zwar keinen C-Kurs ersetzen, jedoch sind die ersten Programmbeispiele anhand der Quellcodes ausführlich erläutert und viele grundlegende C-Programm-Bestandteile wie Präprozessor-Anweisungen, init main, while, Klammern, Konstanten, Zustandsautomat usw. erklärt. Damit und mit einem C-Kurs, den man sehr leicht im Internet findet, kann man bereits die ersten einfachen Programme schreiben. Die mitgelieferte Programmbeispielsammlung gipfelt bereits in eindrucksvollen Anwendungen wie Hinderniserkennung und Fahrt entlang einer Linie.

Die zweite Möglichkeit, der „Robotik Online Code Compiler" (Roboter.CC, Bild 5), erinnert etwas an die IDE des Arduino-Projekts.

Allerdings muss man hier noch eine spezielle Programmer-Software herunterladen, das Open-Source Projekt „RoboDude", und neben der NIBObeelib (USB-Treiber!) auf dem PC installieren (Bild 12). Denn der Compiler von Roboter.CC erzeugt ein XML-basiertes Hex-Format, in dem nicht nur das klassische Hex-Programmfile selbst untergebracht ist, sondern auch alle weiteren Informationen wie Fuse-Bit-Definitionen, EEPROM-Daten, Mikrocontroller-Informationen und solche zum Zielsystem. Der RoboDude-Programmer „sortiert" diese Informationen wieder aus dem Paket und schickt sie nach Norm in das Zielsystem.

Die bereits im Tutorial in Zusammenarbeit mit der AVR-Studio-IDE besprochenen Testprogramme finden sich auch hier im Roboter.CC wieder. Man kann sie, ebenso wie weitere öffentliche Programme, einfach in die IDE laden, ansehen und als fertiges Xhex-, Hex- und Quellcode-File herunterladen.

Hier finden sich sowohl Programme anderer NIBObee-Besitzer als auch solche von nicai-systems, etwa zu den Erweiterungsbausteinen des Roboters, auf die wir noch etwas näher eingehen werden.

Meldet man sich als Mitglied der Community auf der Plattform an, kann man sofort mit dem Erstellen eigener Programme beginnen. Einfach ein neues Projekt anlegen, die gewünschten Einstellungen (Bild 13) vornehmen und den Code in die IDE schreiben (Bild 14). Über „Save & compile" wird der Code (hinter der IDE steckt übrigens der Open-Source-Compiler „avrgcc") compiliert, eventuelle Fehler und das Ergebnis der Arbeit des Compilers gemeldet.

Das Projekt wird unter dem eigenen Log-in oder (wenn so definiert) als öffentliches Projekt gespeichert und kann als Quellcode auch auf dem eigenen Computer per ZIP-File gespeichert werden. Hat man es als nicht öffentlich deklariert, ist es jederzeit nach erneutem Einloggen auch auf Roboter.CC wieder zugänglich, kann auch nachträglich editiert, gelöscht oder veröffentlicht werden.

Bild 13: Die Einstellungen für das eigene Projekt

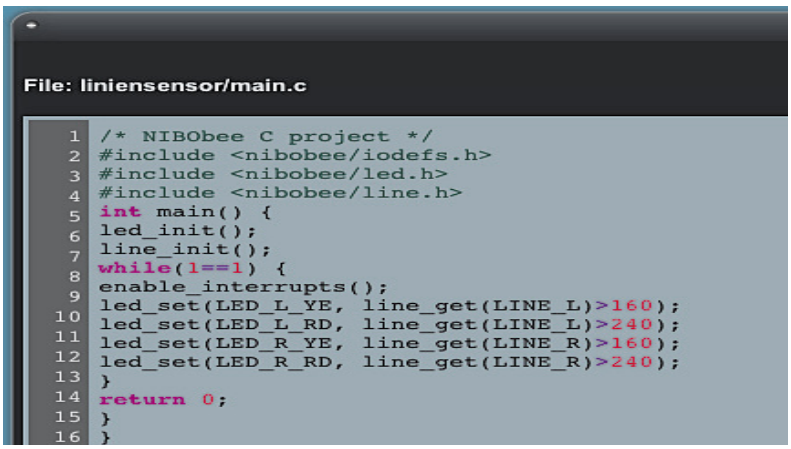

Bild 14: Das C-Programm wird einfach wie in einem normalen Programmeditor Zeile für Zeile geschrieben, die Farbmarkierungen werden automatisch vorgenommen je nach Anweisung, Kommentar usw.

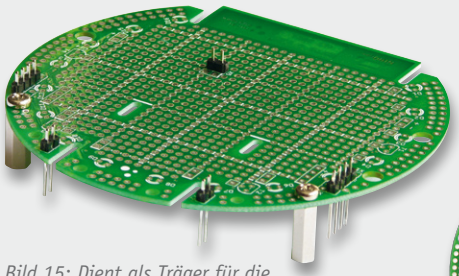

Bild 15: Dient als Träger für die verschiedensten Experimente: die BXT9-Erweiterung.

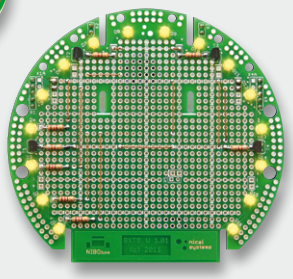

Bild 16: Basiert auf der BXT9-Erweiterung: LED-Lauflicht BKit1.

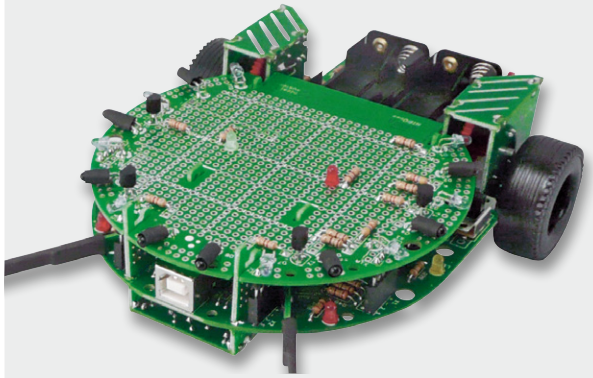

Bild 17: Fast-Rundumblick: Die Hinderniserkennung BKit2 im Einsatz auf dem NIBObee. Die gesamte Verkabelung liegt aus optischen Gründen unter der Platine.

Insgesamt ist Roboter.CC als wirklich interessante ballastfreie Alternative zur herkömmlichen Programmierumgebung zu sehen, freilich mit einigen Grenzen, die aber nur bei speziellen Programmanforderungen erreicht werden – eine tolle Sache, die dem Ziel, Einsteiger an C oder auch Java heranzuführen, wirklich sehr dienlich ist.

Weiterbauen!

Der NIBObee ist ja bereits in der Grundausstattung mit zahlreichen Sensoren bestückt – es geht aber noch mehr! Der Hersteller bietet dazu einiges an Aufrüstmöglichkeiten. Das beginnt bei einer einfachen Hinderniserkennung, geht weiter über eine aufsteckbare Universalplattform (Bild 15), die als Träger für weitere Bauteil-Kits oder eigene Schaltungen dient. Man kann ein Lauflicht (Bild 16) bauen, es gibt ein Remote-Kit, das den Roboter über eine RC5-Fernbedienung steuerbar macht, und ein Kit für die Hinderniserkennung (Bild 17). Wem Rechenleistung und Speicher irgendwann nicht mehr genügen, der kann zum Tuning-Kit mit zwei leistungsfähigeren Prozessoren greifen.

Highlight des Zubehörs und I-Tüpfelchen bei der Aufrüstung des NIBObee ist sicher die Grafikdisplay-Erweiterung (Bild 18). Die kostet zwar genauso viel wie der Roboter-Bausatz selbst, macht ihn aber endgültig zum autonomen Gerät. Denn hier wirkt ein eigener AVR (ATmega88, 8 MHz), der per eigener, integrierter Funktionsbibliothek nicht nur das Display „bedient" und die vier Tasten nebst Anzeige-LEDs verwaltet. Er kommuniziert per Bus mit dem NIBObee-Prozessor und kann diesen befehligen, Abfragen machen usw. So kann man etwa eine komplette Bedienoberfläche kreieren, verschiedene, per Menü abrufbare Aufgaben für den Roboter hinterlegen, Sensoren, z. B. Klimasensoren, abfragen, für die rings um das Display noch Lochrasterflächen bereitstehen. Daneben ist das Display-Kit auch als eigenständige AVR-Applikation betreibbar. Wenn man also den NIBObee gerade nicht benutzt, „schiebt" man eine eigene Software auf den AVR und kann das Kit dann z. B. als kleine Wetterstation an die Wand hängen. Für Show-Zwecke lassen sich sogar Grafiken auf das Display bringen (Bild 19), wie etwa der Namensgeber des kleinen Roboters mit den charakteristischen Fühlern. Details zur Programmierung der Erweiterung finden sich in der Wiki zu jedem Teil des Systems auf der Seite von nicai-systems.

Fazit der ersten Bekanntschaft mit dem NIBObee: mehr als einer von vielen Modell-Robotern, eine perfekte Verbindung zwischen Elektronik-Selbstbau, Lernen einer Programmiersprache und Bereitstellung einer einfachen Programmierumgebung. Dazu kommen die umfangreichen Ausbaumöglichkeiten und ein günstiger Preis, für den man auch kaum einen solchen Fahrroboter „zu Fuß" selbst bauen kann. Zumal man hier die Erfolgsgarantie mitkauft! ELV

Weitere Infos:

[1] Metall-Roboterarm AREXX RA1-PRO, ELV-Best.-Nr. JN-10 18 28
[2] Roboterbausatz nicai-systems NIBO 2, ELV-Best.-Nr. JN-10 21 20
[3] Roboterbausatz nicai-systems NIBObee, ELV-Best.-Nr. JN-10 21 12
Alle Robotik-Bausätze finden Sie unter www.robotik.elv.de

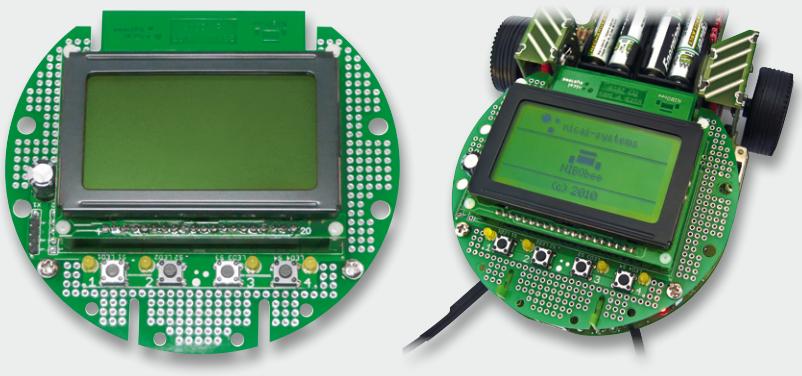

Bild 18: Steigert den Nutzwert des Fahrroboters enorm – die Grafikdisplay-Erweiterung BGX1.

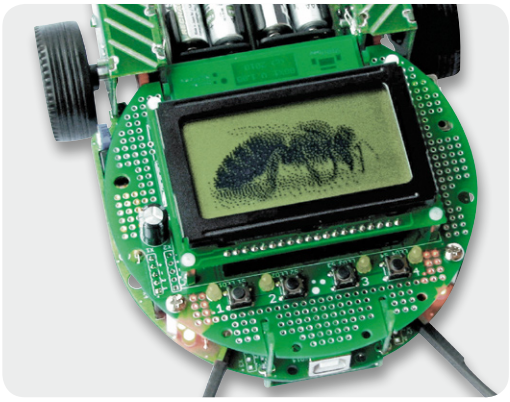

Bild 19: „Wallpaper" mit Namensgeber – auch so etwas kann das kleine Grafikdisplay.