

Praktisches Programmieretool

Arduino Nano als ISP-Programmer

Der übliche Weg, Arduino & Co zu programmieren, führt meistens über die Verbindung des Entwicklungsboards per USB-Kabel direkt an den Rechner. Doch was, wenn man für sein Projekt einen Mikrocontroller nutzen möchte, auf dem beispielsweise noch kein Bootloader aufgespielt ist? Miguel Köhnlein und Michael Gaus nutzen dafür einen Arduino Nano als ISP-Programmer (ISP = In-System-Programming) und werfen zudem noch einen Blick auf das Programmieretool BitBurner.

Vielen Dank!



**Miguel Köhnlein aus Lauterbach
und Michael Gaus aus Schramberg**

haben für ihren Beitrag zum Leserwettbewerb
einen Gutscheincode im Wert von € 200,- erhalten.

Funktionalität

- Arduino Nano als preisgünstigen AVR-ISP-Programmer einsetzen
- Direkt ansteuerbar aus der Arduino-IDE
- Kann verwendet werden, um Arduino Bootloader zu flashen
- Als universelles AVR-ISP-Programmiergerät verwendbar mit dem Tool BitBurner
- 6-poliger Pfostenverbinder, der direkt auf Standard-ISP-Anschluss passt
- Optional kann das Target über den ISP-Programmer mit 5 V versorgt werden

Kurzbeschreibung

Mit einem Arduino Nano lässt sich mit wenig Aufwand ein ISP-Programmer bauen, der entweder direkt von der Arduino IDE oder über das Tool BitBurner angesteuert werden kann.

Beispielsweise kann das Arduino Nano Board aus dem ELVshop verwendet werden [1]. Es sind nur wenige zusätzliche Bauteile erforderlich. Nach Aufspielen eines in der Arduino IDE verfügbaren Beispiel-Sketches (s. u. Arduino Nano als ISP-Programmer flashen) ist der ISP-Programmer bereit für den Einsatz.



Da der Arduino-Nano-ISP-Programmer direkt aus der Arduino IDE ansteuerbar ist, kann er somit auch verwendet werden, um einen Arduino-Bootloader zu flashen.

Der Programmer ist kompatibel mit dem bekannten Programmierwerkzeug AVRdude und kann auch mithilfe des Tools BitBurner über eine grafische Oberfläche bedient werden, ohne dass man sich mit den Kommandozeilenaufrufen von AVRdude vertraut machen muss.

Der Anschluss des zu programmierenden AVR-Controllers erfolgt über einen 6-poligen Pfostenverbinder (Buchse), der direkt auf den 6-poligen Standard-ISP-Anschluss (Stecker) passt, der in der AVR-Welt üblich ist. Über einen Jumper kann ausgewählt werden, ob das Target über den ISP-Programmer mit 5 V versorgt werden soll.

Dadurch kann beispielsweise ein nackter AVR-Mikrocontroller mit wenig Schaltungsaufwand programmiert und in eine Stand-alone-Schaltung ohne ISP-Anschluss eingebaut werden.

Mit dem ISP-Programmer können auch manche ELV Bausätze mit AVR-Mikrocontrollern, bei denen der Quellcode bereitgestellt wird, mit angepasster eigener Firmware neu programmiert werden.

Über den Jumper JP1 wird eingestellt, ob die 5-V-Versorgungsspannung zum Pin 2 des Target-Connectors geführt wird. Die Jumpereinstellungen sind wie folgt:

- JP1 offen: keine Versorgungsspannung, Target muss separat versorgt werden
- JP1 gesteckt: 5-V-Versorgungsspannung des Arduino Nano wird zum Target geführt

Hinweis: Wenn das Target über eine eigene Spannungsquelle versorgt wird und diese mit J1 Pin 2 verbunden ist, darf der Jumper JP1 nicht gesteckt werden!

Über den Jumper JP2 kann der Resetpin des Arduino Nano mit einem Kondensator gegen GND beschaltet werden. Die Jumpereinstellungen sind wie folgt:

- JP2 offen: kein Kondensator am Resetpin
- JP2 gesteckt: Am Resetpin ist ein 10-µF-Kondensator gegen GND vorhanden

Hinweis: Bei der Programmierung des Arduino Nano darf JP2 nicht gesteckt sein, da ansonsten der Bootloader keinen Reset vor der Programmierung des Sketches auslösen kann.

Hardware

Bild 1 zeigt den Schaltplan des ISP-Programmers. J1 ist der 6-polige ISP-Steckverbinder zum Anschluss des zu programmierenden AVR-Mikrocontrollers, die Pinbelegung sieht folgendermaßen aus:

Pinbelegung	Pin	Signal
	1	MISO
	2	+5 V (nur wenn JP1 gesteckt ist)
	3	SCK
	4	MOSI
	5	/Reset (für den zu programmierenden AVR-Controller)
	6	GND

Der Jumper wird nur benötigt, falls sich der Arduino Nano beim Betrieb als ISP-Programmer beim Connect-Vorgang resetten sollte und dadurch keine Kommunikation zustande kommt. Im Normalfall sollte ein Betrieb ohne Jumper möglich sein.

Die LEDs dienen als Statusanzeige und können optional entfallen, falls weitere Bauteile eingespart werden sollen.

Die Bedeutung der LEDs ist wie folgt:

- LED1 grün: Heartbeat-LED, wird zyklisch auf- und abgedimmt
- LED2 rot: Error, es ist ein Fehler bei der ISP-Programmierung aufgetreten
- LED3 gelb: leuchtet während der ISP-Programmierung

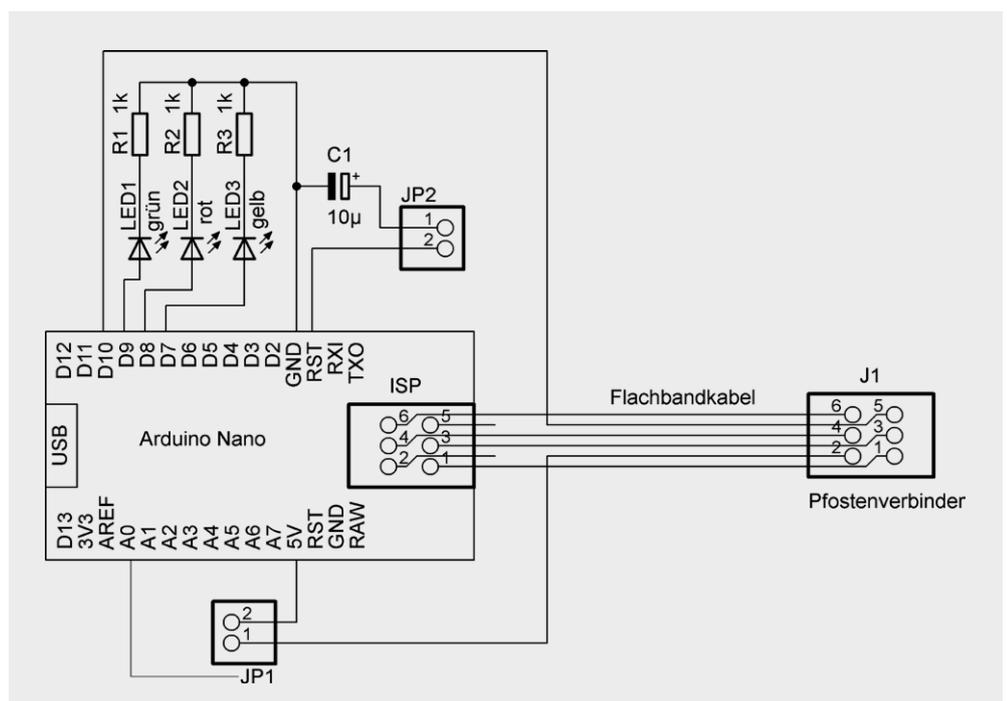


Bild 1:
Schaltplan des
ISP-Programmers

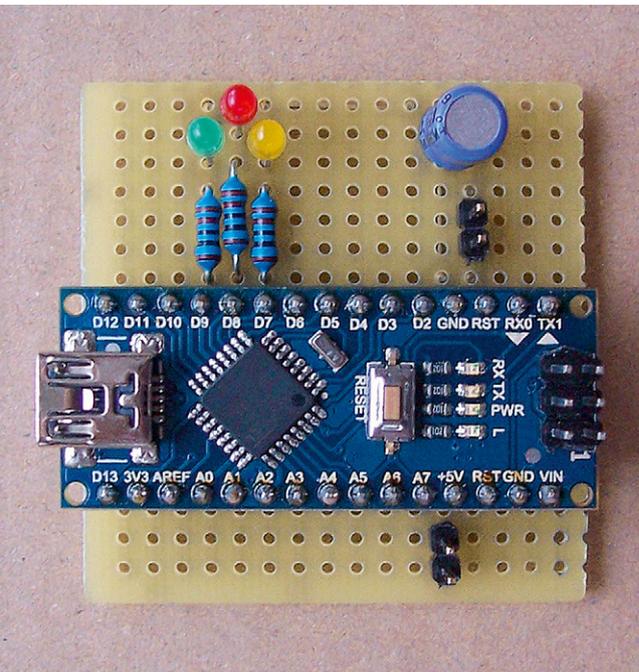


Bild 2: Arduino Nano mit den Bauteilen auf einer Lochrasterplatine

Aufbau

Die Schaltung kann auf einem Stück Lochrasterplatine mit Lötstellen im 2,54-mm-Raster aufgebaut werden. Nebeneinanderliegende Verbindungen können mit Lötzinnbrücken einfach hergestellt werden. Für längere Strecken werden Drahtstücke auf die Unterseite aufgelötet. Der Arduino Nano wird in zwei passende jeweils 15-polige Buchsenleisten gesteckt und ist dadurch austauschbar, um ihn bei Bedarf auch für andere Zwecke verwenden zu können (Bild 2).

Das Flachbandkabel wird auf die beiden 6-poligen Pfostenverbinder gecrimpt (oder per Schraubstock aufgequetscht). Es muss darauf geachtet werden, dass die farbig markierte Ader jeweils auf Pin 1 kontaktiert wird. Anschließend werden wie auf dem Foto gezeigt auf der linken Seite des Kabels, die später auf den ISP-Stecker des Arduino Nano aufgesteckt wird, die Adern 2 und 5 aufgetrennt (also jeweils die zweite Ader beider Seiten) (Bild 3). Hierzu kann man mit einem Skalpell oder scharfen Messer vorsichtig zwischen die jeweiligen Adern ritzen und dann mit einem Seitenschneider die Adern durchknipsen. Danach werden diese abisoliert, verdreht und verzinkt und können dann direkt in die Lochrasterplatine am Rand eingelötet werden und von dort mit Drahtstücken zum jeweiligen Schaltungsteil verbunden werden (Bild 4).

Arduino Nano als ISP-Programmer flashen

Folgende Schritte sind notwendig, um über die Arduino IDE den Arduino Nano als ISP-Programmer zu konfigurieren:

- Auf der zuvor gezeigten ISP-Programmer-Platine den USB-Anschluss des Arduino Nano mit dem PC verbinden
- Jumper JP2 darf nicht gesteckt sein
- Sketch „ArduinoISP“ öffnen unter Beispiele → 11.ArduinoISP → ArduinoISP



Bild 3: Die Adern 2 und 5 werden aufgetrennt und zu den Jumpers geführt.

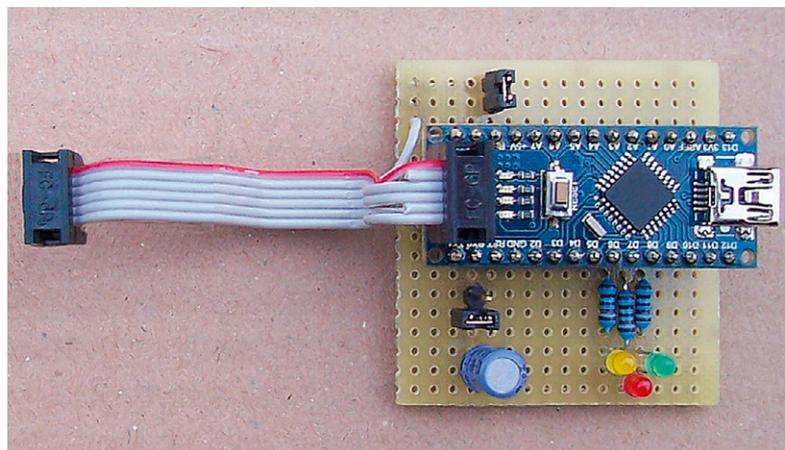


Bild 4: Aufbau mit Flachbandkabel

- Werkzeuge → Board: „Arduino Nano“
- Werkzeuge → Prozessor: „ATmega328P (Old Bootloader)“ bzw. bei neueren Nanos nur „ATmega328P“ (am besten durch Ausprobieren herausfinden)
- Werkzeuge → Port: passenden COM-Port des Arduino Nano auswählen
- Sketch → Überprüfen/Kompilieren
- Sketch → Hochladen
- Nun sollte die grüne LED1 auf- und abdimmern als „Heartbeat“-Anzeige
- Die beiden LEDs 2 und 3 sollten nun aus sein
- Der Sketch ArduinoISP kann wieder geschlossen werden

Der Arduino Nano kann nun als ISP-Programmer des Typs „Arduino as ISP“ verwendet werden.

Arduino-Bootloader in fabrikneuen ATmega328P flashen

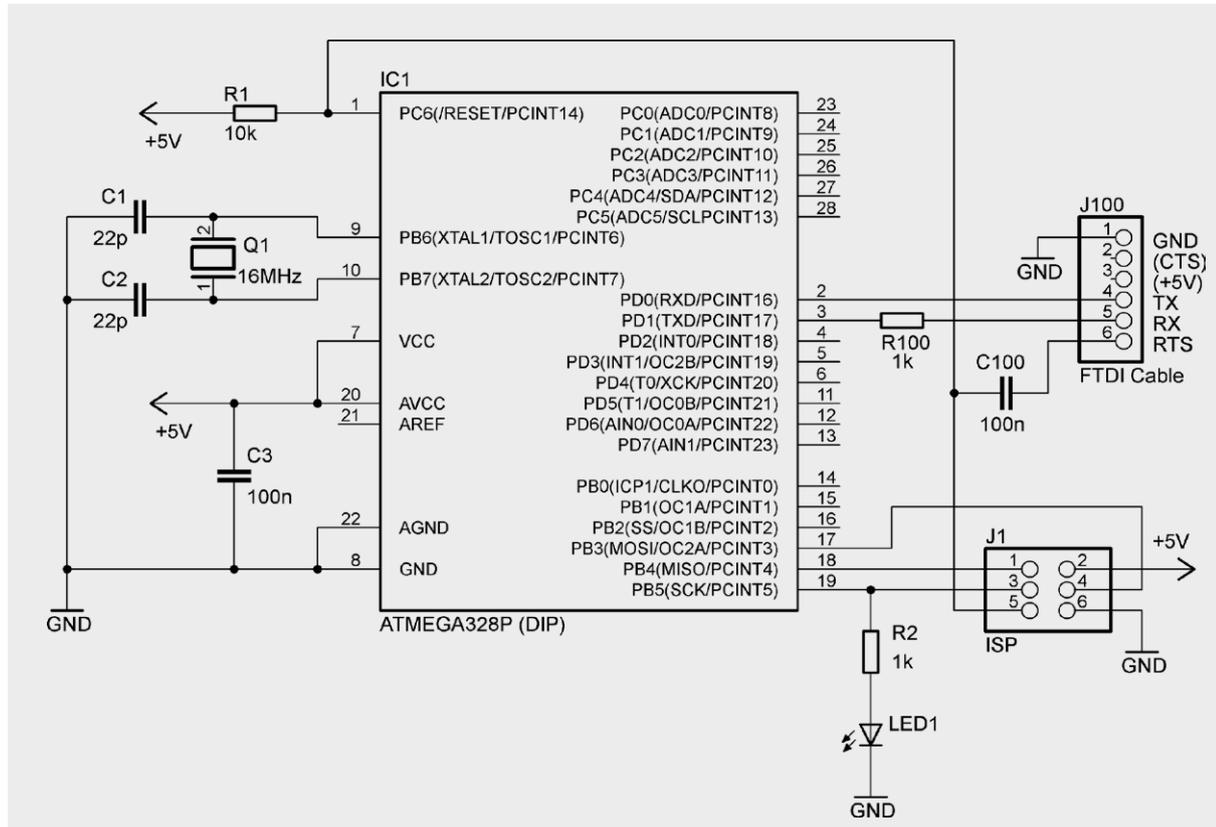
Ein fabrikneuer ATmega328P soll mit dem passenden Bootloader geflasht werden, um ihn dann per Arduino IDE programmieren und als Stand-alone-Anwendung mit Minimalbeschaltung ohne die komplette UNO-Hardware einsetzen zu können. Hierzu verwenden wir als Hardwarebasis zum Flashen unsere Stand-alone-Version oder auch die Version Arduino UNO R3 DIP [3].

Der vorhandene Prozessor auf diesem UNO-Board ist gesockelt und kann somit leicht ausgetauscht werden. Er wird nun durch den fabrikneuen ATmega328P ersetzt (nur bei ausgeschaltetem Board tauschen). Nach dem Anschluss an den USB-Port des PCs sind folgende Schritte in der Arduino IDE erforderlich, um den UNO-Bootloader zu flashen:

- Werkzeuge Board → Gewünschtes Targetboard auswählen, also Arduino UNO
- Werkzeuge → Programmer: „Arduino as ISP“ (auf genaue Bezeichnung achten!)
- Werkzeuge → Port: COM-Port des Arduino-Nano-ISP-Programmers auswählen
- Jumper JP1 entfernen, da der Arduino UNO separat über seinen USB-Anschluss mit Spannung versorgt wird
- ICSP-Anschluss des Arduino UNO mit J1 des Arduino-Nano-ISP-Programmers verbinden, auf richtige Polung achten



Bild 5: Schaltplan
Arduino Stand-alone



- Werkzeuge → Bootloader brennen
- Nun sollte die Board-LED des Arduino UNO blinken als Zeichen dafür, dass der Bootloader aktiv ist, da kein Sketch im Flash vorhanden ist
- Nachdem der Bootlader auf dem Arduino UNO vorhanden ist, können wir einen Sketch programmieren
- Als Test öffnen wir den Demosketch unter Beispiele → 01.Basics → Blink
- Werkzeuge → Port: passenden COM-Port des Arduino UNO auswählen
- Sketch → Überprüfen/Kompilieren
- Sketch → Hochladen
- Nun sollte die Board-LED des Arduino UNO im Sekundentakt an- und ausgehen

Nun kann der ATmega328P aus dem Sockel entfernt und in der Stand-alone-Anwendung eingebaut werden. Hierzu ist die Beschaltung mit einem 16-MHz-Quarz mit passenden Anschwingkondensatoren sowie ein 100-nF-Abblockkondensator zwischen Vcc und GND notwendig.

Bild 5 zeigt den Schaltplan des Arduino-Stand-alone. Die Bauteile R100, C100 und J100 sind nur erforderlich, wenn von der Arduino IDE aus ein Sketch ohne zusätzlichen ISP-Programmer auf den ATmega328P geladen werden soll. In diesem Fall benötigt man einen USB-Seriell-Wandler, z. B. ein FTDI-Kabel.

Programmiertool BitBurner

Mithilfe des Tools BitBurner ist der ISP-Programmer als universelles AVR-ISP-Programmiergerät verwendbar. Das Tool bietet eine GUI für die gewünschten Einstellungen und ruft für den eigentlichen Programmiervorgang das bekannte Tool AVRdude auf.

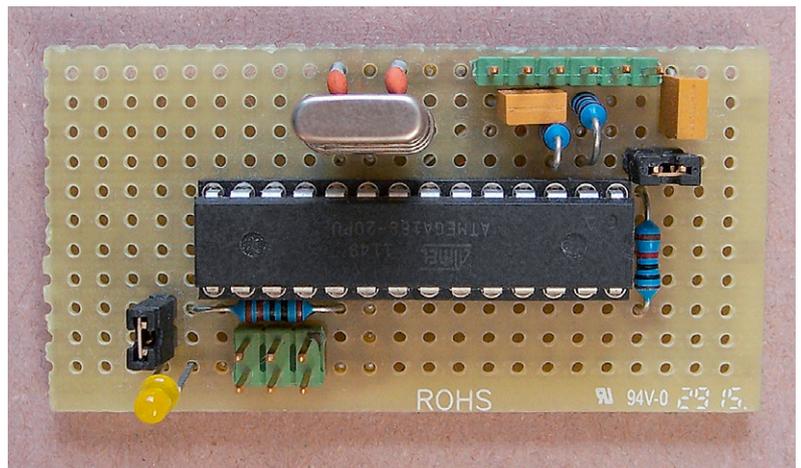


Bild 6: Stand-alone-Version des ATMEGA328 (Arduino UNO)

BitBurner kann auf folgender Seite heruntergeladen werden [4]. Damit die Kommunikation zu unserem ISP-Programmer funktioniert, sind auf der Hauptseite im Reiter „AVRdude“ folgende Einstellungen erforderlich:

- AVR Device: gewünschten Target-Controller auswählen, hier im Beispiel ATmega168
- Programmer: Atmel STK500 Version 1.x firmware (stk500v1)
- COM-Port: passenden COM-Port des Arduino Nano ISP-Programmers auswählen
- Die Angaben im Feld „AVRdude Arguments“ müssen nicht manuell eingegeben werden, sondern werden durch die Konfiguration unter Tools → Settings im nächsten Schritt automatisch eingetragen (Bild 7)

Bei Tools → Settings muss im Reiter „AVRdude“ die Baudrate auf 19.200 Bd eingestellt werden. Dadurch wird dann automatisch das Argument „-b 19200“ beim Aufruf von AVRdude übergeben, was im Bild 7 im Feld „AVRdude Arguments“ zu sehen ist (Bild 8).

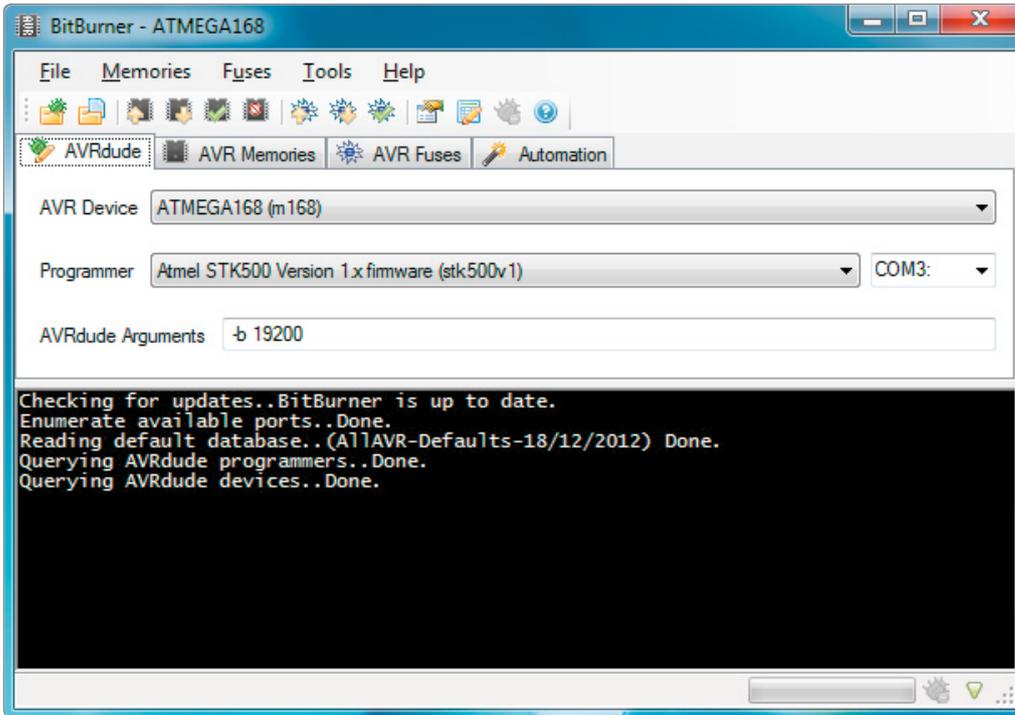


Bild 7: Bitburner: Einstellungen

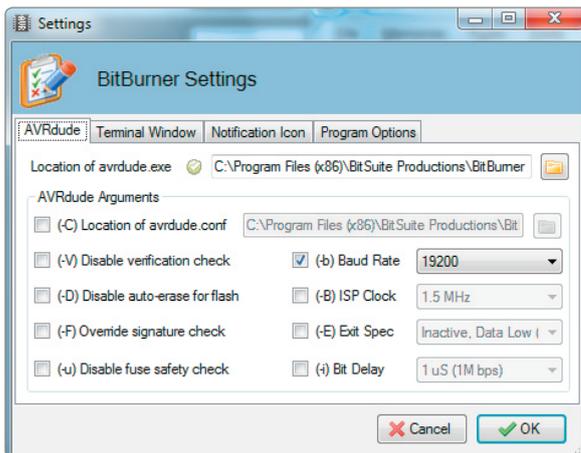


Bild 8: Einstellen der Baudrate auf 19.200 Bd

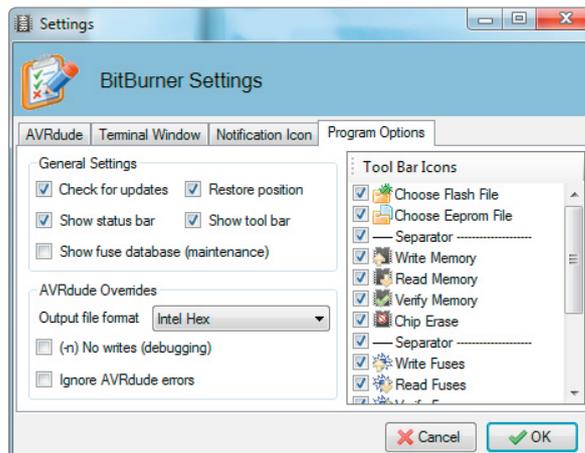


Bild 9: Auswahl des Output-Formats

Bei Tools → Settings kann im Reiter „Program Options“ bei „Output file format“ das gewünschte Dateiformat ausgewählt werden. Üblich ist „Intel Hex“, für reine Binärdateien steht „Raw binary“ zur Verfügung (Bild 9).

Auf der Hauptseite im Reiter „AVR Memories“ können Flash und/oder EEPROM des AVR-Controllers gelesen oder geschrieben werden (Bild 10).

Auf der Hauptseite im Reiter „AVR Fuses“ können die Fuses und/oder Lockbits gelesen oder geschrieben werden. Die Fuses werden im Klartext angezeigt, sodass kein weiteres Tool zur Umrechnung der Fuse-Bits erforderlich ist (Bild 11). **ELV**

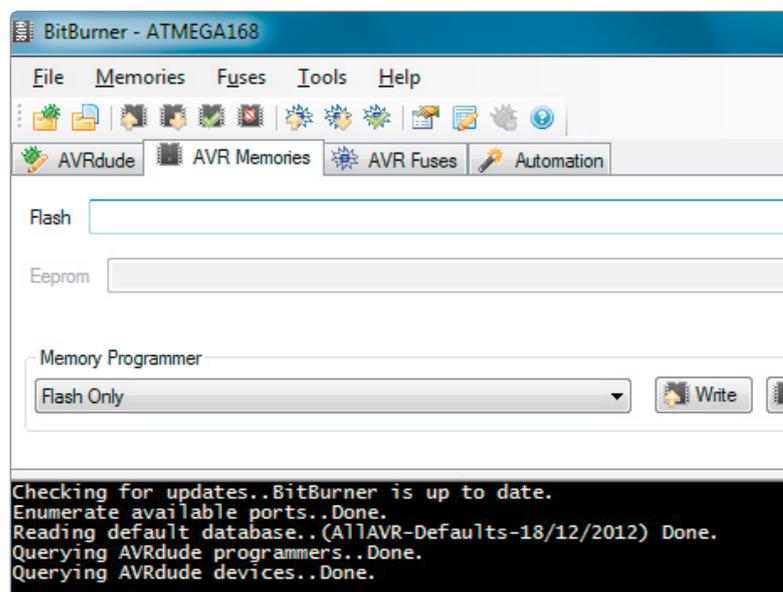


Bild 10: BitBurner bietet die Möglichkeit zum Auslesen des Speichers bzw. EEPROM.

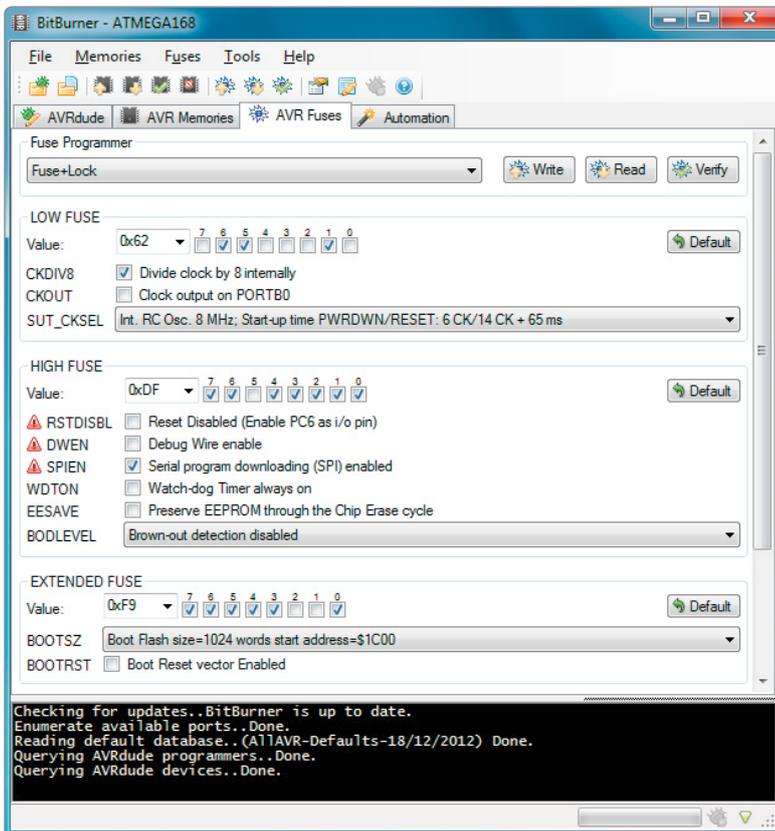


Bild 11: Setzen bzw. Lesen der Lock-/Fuse-Bits

Stückliste

Arduino Nano	Arduino
Buchsenleiste für Arduino Nano	Arduino
Widerstand 1 kΩ	R1, R2, R3
LED, grün, 3 mm	LED1
LED, rot, 3 mm	LED2
LED, gelb, 3 mm	LED3
Kondensator 10µF	C1
Stiftleiste	JP1, JP2
Jumper	JP1, JP2
Pfostenverbinder PV6	ISP, J1
Flachbandleitung	Kabel
Lochrasterplatine mit Lötunkten	Platine

Hier finden Sie alle Produkte:
Alle Produkte für dieses Projekt finden Sie unter nebenstehendem QR-Code.



Weitere Infos:

- [1] JOY-iT Arduino Nano Board:
Bestell-Nr. 250005
- [2] Komplettbausatz Würfelspiel –
Meiern WM100: Bestell-Nr. 91580
- [3] JOY-iT Arduino kompatibles Board
UNO R3 Dip Version: Bestell-Nr. 122123
- [4] sourceforge.net/projects/bitburner/

Alle Links finden Sie auch online unter de.elv.com/elvjournal-links

Unser Leserwettbewerb – teilen Sie Ihr Lieblingsprojekt!

Machen Sie mit!

Jede veröffentlichte Anwendung belohnen wir mit einem Gutscheincode im Wert von € 200,-*.



Das umfangreiche Angebot von ELV Haustechniksystemen, Produkten und Bausätzen bietet für viele Leser den Ausgangspunkt für eigene kreative Ideen. Haben auch Sie ein Projekt entwickelt, das andere Leser interessieren könnte?

Alles, was nicht gegen Gesetze oder z. B. VDE-Vorschriften verstößt, ist für uns interessant. Schreiben Sie uns, fotografieren Sie Ihr Projekt, berichten Sie von Ihren Erfahrungen und Lösungen. Teilen Sie Ihre fantasievolle Idee mit den Lesern des ELVjournals!

Die interessantesten Anwendungen werden redaktionell bearbeitet und im ELVjournal mit Nennung des Namens vorgestellt.

Die Auswahl der Veröffentlichungen wird allein durch die ELV Redaktion ausschließlich nach Originalität, praktischem Nutzen und realisierter bzw. dokumentierter Ausführung vorgenommen, es besteht kein Anspruch auf Veröffentlichung, auch bei themengleichen Lösungen. **Der Rechtsweg ist ausgeschlossen.** Für Ansprüche Dritter, Beschädigung und Verlust der Einsendungen wird keine Haftung übernommen. Alle Rechte an Fotos, Unterlagen usw. müssen beim Einsender liegen. Die eingesandten Unterlagen und Aufnahmen verbleiben bei der ELV Elektronik AG und können von dieser für Veröffentlichungen und zu Werbezwecken genutzt werden.

* Der Einsender der veröffentlichten Anwendung erhält einen Gutscheincode zur einmaligen Nutzung im Wert von € 200,-. Der Gutscheincode wird mit einer Bestellung verrechnet – ein etwaiger Restbetrag verfällt. Bei Rückabwicklung des Kaufvertrages oder eines Teils hiervon wird der gewährte Gutscheincodebetrag vom zu erstattenden Kaufpreis abgezogen, sofern durch die Ausübung des Widerrufsrechts und der Rückabwicklung der Gesamtwarenwert von €200,- unterschritten wird. Auszahlung/Verrechnung mit offener Rechnung sowie Gutschrift nach Widerruf sind nicht möglich. Der Gutscheincode ist nicht mit anderen Aktionen kombinierbar.



Per Email
leserwettbewerb@elv.com



Per Post
ELV Elektronik AG
ELVjournal Leserwettbewerb
26787 Leer