

**Kurzanleitung zur
Installation und ersten Nutzung
des ISP Programmieradapters
Atmel MK2 / Diamex All AVR**

**V 1.11
2. Mai 2015**



© 2015 by Peter Küsters

Dieses Dokument ist urheberrechtlich geschützt. Es ist nicht gestattet, dieses Dokument zur verändern und komplett oder Teile daraus ohne schriftliche Genehmigung von uns weiterzugeben, es zu veröffentlichen, es als Download zur Verfügung zu stellen oder den Inhalt anderweitig anderen Personen zur Verfügung zu stellen.

Inhaltsverzeichnis

INHALTSVERZEICHNIS	2
ISP KABEL (6ER / 10ER), PDI KABEL	4
UNTERSCHIED ATMEL ISP MKII UND DIAMEX ALLAVR	5
DIAMEX ALLAVR PROGRAMMIERADAPTER	6
SPANNUNGSVERSORGUNG DES MIKROCONTROLLERBOARDS ÜBER DEN PROGRAMMIERADAPTER	6
JUMPER.....	7
INSTALLATION VON ATMEL STUDIO	9
DER ERSTE PROGRAMMIERVORGANG:	11
1. PROGRAMMIERADAPTER AUSWÄHLEN	11
2. AUSWAHL DES GENUTZTEN MIKROCONTROLLERS	12
3. PRÜFEN DER VERBINDUNG	13
4. AUSWAHL DER ISP PROGRAMMIERFREQUENZ.....	14
5. PROGRAMMIERVORGANG	16
6. FUSES / FUSE-BITS.....	18
7. LOCK BITS	19
SONSTIGE HILFE UND INFORMATIONEN	19
ANHANG: AVR MKII / ALLAVR UND BASCOM	20
ANHANG: FIRMWARE UPDATE	21
ANHANG: UNTERSCHIEDE BEI ATMEL STUDIO 5 / 6	22
TYPISCHE PROBLEME:	24
ICH KANN KEINE VERBINDUNG ZUM PROGRAMMER AUFBAUEN.	24
DIE LEUCHTDIODE IM INNERN DES PROGRAMMERS SCHALTET IMMER MAL WIEDER AB UND ICH MUSS DEN PROGRAMMER VOM USB PORT TRENNEN UND WIEDERVERBINDEN.....	24
ICH KANN KEINE VERBINDUNG ZU MEINEM CONTROLLER AUFNEHMEN. BEIM DRÜCKEN VON „READ“ BEI „DEVICE SIGNATURE“ ERSCHEINT NUR ----	24
BEIM PROGRAMMIEREN ERHALTE ICH IMMER MAL WIEDER EINE DIALOGBOX MIT FEHLERN, NACH EIN PAAR MAL DRÜCKEN AUF „OK“ ZUM WIEDERHOLEN GEHT ES DANN.....	24
ICH HABE IN DEN FUSES ETWAS VERSTELLT UND NUN BEKOMME ICH KEINEN KONTAKT MEHR ZUM CONTROLLER.	24
ICH HABE KEINE FUSES VERSTELLT, TROTZDEM BEKOMME ICH KEINEN KONTAKT MEHR ZUM MIKROCONTROLLER.	24
DIE LED LEUCHTET, ICH KANN ABER IMMER NOCH NICHT KONTAKT ZUM PROGRAMMER AUFNEHMEN ODER ICH BEKOMME KEINEN KONTAKT MEHR ZUM CONTROLLER.....	25
ICH HABE MIR EIN EIGENES BOARD GEBAUT UND ICH KANN ES NICHT MIT DEM PROGRAMMIERADAPTER ANSPRECHEN.	25
KONTAKT:	26

Herzlichen Glückwunsch zum AllAVR bzw. MKII (MK2) ISP Programmieradapter.

Vorab noch ein Plädoyer für diesen Programmieradapter: Die MK2 bzw. kompatiblen Programmieradapter sind sicherer und schneller als die üblichen, preiswerteren USB/RS232 Programmieradapter. Dieser Typ prüft zudem sämtliche Leitungen und informiert bei falscher Verdrahtung darüber, welche der Leitungen nicht korrekt erscheinen. Fehlprogrammierungen, wie sie älteren Programmieradaptern hin und wieder zu beklagen waren (z.B. zerschossene Fuses ohne dass man diese überhaupt beschrieben hat), sind hier nicht mehr beobachtet worden. Das Preis-/Leistungsverhältnis ist als exzellent zu bezeichnen.

Dieser MKII (kompatible) Programmieradapter unterstützt verschiedene Controllertypen aus verschiedenen Familien (AVR, AVR32 und XMeta) und unterstützt die Programmiermodi ISP, TPI und PDI.

Ob Sie einen hochwertigen Programmieradapter besitzen, erkennen Sie i.d.R. an der Komplexität: der MKII und der Diamex ALLAVR (nicht zu verwechseln mit dem billigen und langsamen Diamex USB ISP) besteht aus 5 bzw. 4 Chips. Einfache und wesentliche langsamere Programmieradapter bestehen i.d.R. nur aus einem oder zwei Chips.

Die nachfolgenden Seiten sollen Ihnen den Einstieg erleichtern – mehr nicht. Für Antworten auf weitergehende Fragen zur Funktionalität oder bei Problemen suchen Sie bitte die entsprechenden Hilfen beim Hersteller oder nutzen eines der vielen Internet Foren.

Das Paket beinhaltet den Programmieradapter mit ISP Kabel, Handbuch und ein USB Kabel.

Dieser Programmieradapter wird mit dem beigegefügteten Kabel an den USB Anschluss des PCs angeschlossen, das andere ISP-Flachbandkabel wird mit Ihrem Atmel-Mikrocontroller verbunden.

Zur Programmierung ist das Atmel Studio notwendig, welches Sie sich bitte hier downloaden: <http://www.atmel.com/tools/atmelstudio.aspx>

Zur Installation von AVR Studio erhalten Sie auf den folgenden Seiten einige Informationen.

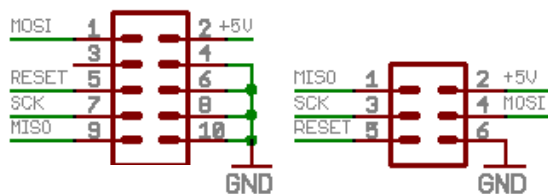
Atmel Studio ist eine Entwicklungsumgebung von Atmel, die Sie nutzen können, es aber nicht müssen. Sofern Sie in Ihrer gewohnten Umgebung arbeiten möchten, so nutzen Sie Ihre gewohnte Entwicklungsumgebung (z.B. WinAVR oder Bascom) und nutzen vom Atmel Studio lediglich das ISP Programmierwerkzeug zum Übertragen der Programmdatei zum Mikrocontroller.

ISP Kabel (6er / 10er), PDI Kabel

Wenn Sie es von anderen ISP Geräten gewohnt waren, ein 50cm oder 1 Meter langes Flachbandkabel vom Programmieradapter zum Controller legen zu können und das beigelegte Kabel verlängern möchten: **Vergessen Sie das ganz schnell.** Dieser Programmieradapter kann die Daten wesentlich schneller einprogrammieren – bei den hohen Frequenzen bzw. der hohen Flankensteilheit dürfen Sie kein längeres Kabel nutzen. Ansonsten laufen Sie in Gefahr, dass sich während des Programmiervorgangs durch Reflektionen und/oder Störungen u.U. die Fuses in Ihrem Controller verstellen und Sie auf diesen nicht mehr zugreifen können. Länger als 20cm darf das ISP Kabel zwischen Programmiergerät und Mikrocontroller nicht mehr sein.

Sie erkennen am MKII Programmiergerät einen **6-poligen** Stecker. Dies ist der neuere ISP Standard von Atmel. Daneben gibt es noch den 10-poligen Standard, bei dem sich mehr Mas-seadern im Kabel befinden.

Die Beschaltung gilt wie folgt:



Anmerkung 1: Vorsicht Falle bei der Erstellung eigener Platinen: Sie erkennen an der Belegung oben die Bezeichnung MOSI/MISO sowie SCK und Reset. MOSI/MISO ist wohl für sehr viele Atmel Controller korrekt, aber nicht für alle. Ein ATMega64, ATMega128, AT90CAN128 oder 2560/2561 z.B. nutzt statt der Ports BI/B2 (die laut Datenblatt MO-SI/MISO sind) die Ports E0 und E1 für den Anschluss des ISP Programmieradapters. In diese Falle sind schon viele (auch der Autor) versehentlich getappt.

Anmerkung 2: Beim Diamex Programmieradapter liegt am 10-poligen Stecker die Masse nur an Pin 10 an. Pin 4, 6, 8 sind nicht mit Masse verbunden. Es schadet aber nichts (im Gegenteil ist es sogar hilfreich), wenn diese Pins auf Ihrem Controllermodul mit Masse miteinander verbunden sind.

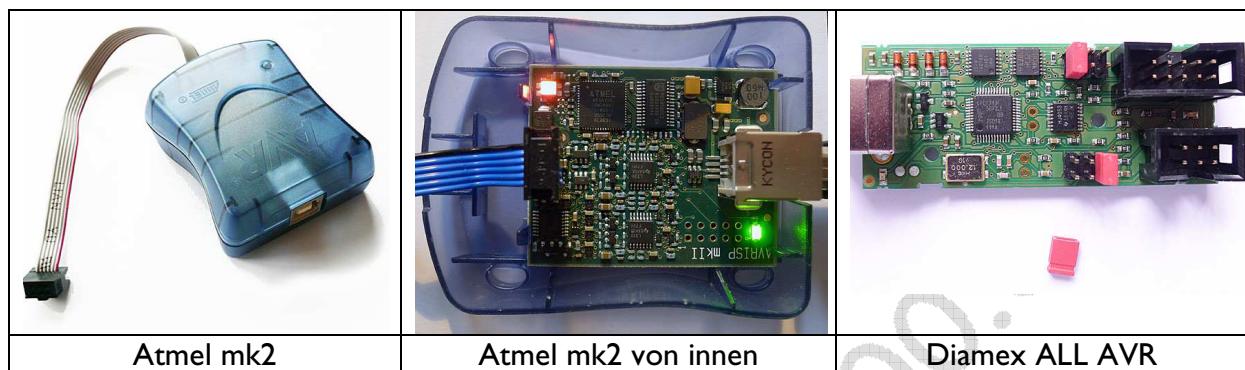
Anmerkung 3: Auf Pin 3 des 10-poligen Stecker wird beim Diamex Programmieradapter während des Programmiervorgangs ein 500 Khz Signal ausgegeben. Damit lassen sich vermeintlich defekte Controller wiederbeleben. Mehr dazu finden Sie auf Seite 8.

XMega-Controller werden mit dem PDI Standard programmiert, was Sie mit diesem Programmieradapter ebenfalls können. Hier gibt es nur ein 6-poliges standardisiertes Kabel. Auch dieses PDI-Kabel darf max. 20 cm lang sein, da die XMega mit hoher Geschwindigkeit programmiert werden.

Unterschied Atmel ISP mkII und Diamex ALLAVR

Für den Programmiervorgang von Atmel Controller können wir nur zwei Programmieradapter empfehlen:

- Den originalen Atmel ISP mk2 (oder mkII) Programmieradapter.
- Den Diamex ALLAVR Programmieradapter, welcher kompatibel zum Original-Atmel ist.



Beide Adapter haben Vor- und Nachteile:

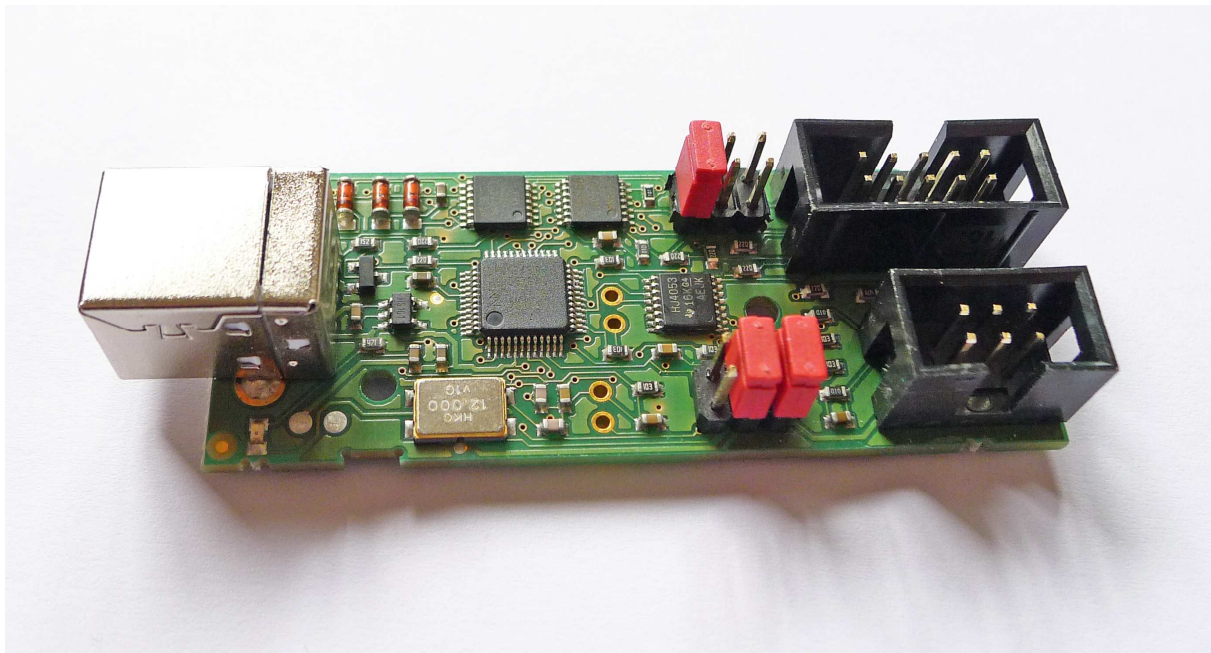
	Atmel ISP mk2	Diamex ALLAVR
Preis	○	++
Inkl. Gehäuse	+	-
Größe	○	+
Geschwindigkeit	++	++
Auf Wunsch Spannungsversorgung des Mikrocontrollers über den Programmieradapter	-	+ (manuelle Wahl von 3,3 & 5,0 Volt)
Taktausgang (zum Retten ‚verfuster‘ Mikrocontroller)	-	++
Umschaltung von ISP zu PDI	++ (automatisch)	○ (muss manuell geschehen)
6er und 10er ISP Adapterkabel inklusive	- (nur 6er ISP)	+ (6er und 10er ISP)

Der originale Atmel Programmieradapter (rechts) bietet keine Optionen, hingegen der ALLAVR Adapter verschiedene Optionen anbietet.

Auf den folgenden drei Seiten gehen wir auf die Möglichkeiten des ALLAVR ein. Besitzer eines originalen Atmel mk2 Programmiers adaptors überspringen diese und können direkt zu Seite 9 weitergehen.

Diamex ALLAVR Programmieradapter

Dem Programmieradapter liegt ein gedrucktes Handbuch bei (welches auch bei uns im Shop als PDF Datei heruntergeladen werden kann), weshalb wir hier nicht im Detail auf ihn eingehen, sondern nur ein paar praxisgerechte Hinweise sowie Informationen über die Jumper-Optionen geben.

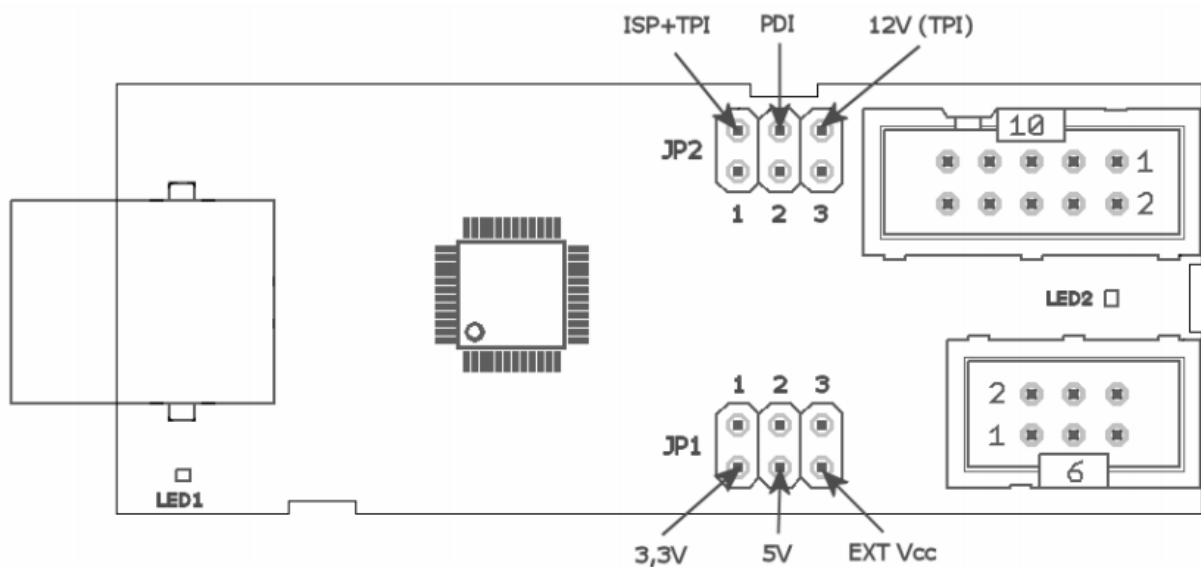


Spannungsversorgung des Mikrocontrollerboards über den Programmieradapter

Laut Handbuch kann der Programmieradapter Schaltungen mit einem Strombedarf von bis zu 50mA versorgen. In der Praxis sind auch etwas höhere Ströme problemlos möglich (bis ca. 80mA). Sobald eine Schaltung einen zu hohen Strom benötigt, stürzt der Programmieradapter beim Anschluss an die ISP/PDI Schnittstelle ab und muss zum Neustart kurz vom USB Bus getrennt werden (man erkennt dies am Status der LED links unten).

Jumper

Die Jumper des Programmieradapters werden in der Regel nicht häufig geändert. In der Praxis geschieht dies nur bei Änderungen der Versorgungsspannung oder beim Wechsel des Controllers von ATmega zu Xmega.



JP1-1: für 3,3 Volt Logiksignale wenn geschlossen

JP1-2: für 5,0 Volt Logiksignale wenn geschlossen

JP1-3: wenn geschlossen, versorgt der Programmieradapter das μ C-Board mit der gewählten Spannung.

Stecken Sie niemals einen Jumper gleichzeitig auf 1 und 2. Die könnte den Programmieradapter beschädigen!

JP2-1: der 6-polige Adapter wird für ISP genutzt wenn geschlossen

JP2-2: der 6-polige Adapter wird für PDI (XMega) genutzt wenn geschlossen

JP2-3: Spezialfall, normalerweise nicht zu nutzen (siehe Handbuch)

Auf JP2 darf immer nur ein Jumper stecken!

Retten eines „defekten“ Controllers

Früher oder später werden Sie einen AVR Mikrocontroller (ATMega, nicht XMega) dadurch verlieren, dass Sie versehentlich falsche Fuse-Einstellungen verwenden und brennen. Der Controller ist dann nicht mehr ansprechbar und vermeintlich defekt.

Dies muss noch nicht einmal absichtlich geschehen. Durch Störungen oder statische Entladung kann es passieren, dass während des Programmiervorgangs korrekter Daten ein Bit kippt und somit etwas falsche einprogrammiert wird.

Tatsächlich gibt es Fuse-Bit Einstellungen (z.B. das Abschalten des ISP Programmierinterfaces oder die Umschaltung des Reset-Ports zu einem „normalen“ Port), die dazu führen, dass der Controller wirklich für immer unbrauchbar wird. Allerdings zeigt die Erfahrung, dass in 90% aller Fälle die Fuse-Bits für den Systemtakt betroffen sind. Der Controller ist dann nicht mehr ansprechbar, da er einen externen Takt erwartet und selbst einen evtl. angeschlossenen Quarz nicht zum Schwingen bringt.

Durch das temporäre Anlegen eines externen Takts an XTALI ist der Controller wieder ansprechbar und die Fuse-Bits können auf die korrekten Werte eingestellt werden.

Der Diamex Programmieradapter bietet hier praktischer Weise einen 500 Khz Ausgang, der für genau diesen Zweck vorgesehen ist.

Vorgehensweise:

1. Verbinden Sie Pin 3 des 10-poligen Programmieradapters (siehe auch das Diamex-Handbuch) mit dem Port XTALI des Mikrocontrollers. Ob Sie zum Lesen bzw. Programmieren das 10-polige oder das 6-polige Kabel nutzen ist egal.
2. Starten Sie das Programmierool – die ISP Programmierfrequenz muss auf 125 Khz eingestellt sein. Der Controller sollte nun ansprechbar sein.
3. Lesen Sie die Werte der Fuse-Bits aus und korrigieren Sie sie. Tipp: wenn Sie einen externen Quarz angeschlossenen haben, ist i.d.R. die allerletzte (unterste) Option der Liste die korrekte Einstellung. Schreiben Sie die Fuses zurück.
4. Der Controller sollte nun wieder ohne den externen (Rettungs)Takt ansprechbar sein.

Hinweis: Der 500 Khz Takt steht nur während des Lese- bzw. Programmiervorgangs zur Verfügung. Vorher und nachher liegt er nicht an und ist somit dann auch nicht messbar/nutzbar.

Wenn Sie keinen Diamex ALLAVR Programmieradapter besitzen, können Sie einen solchen Takt zur Rettung auch z.B. über einen anderen Mikrocontroller oder über einen Funktionsgenerator erzeugen und anlegen.

Installation von Atmel Studio

Für den Programmieradapter mk2 / ALLAVR benötigen Sie das Programmtool, welches sich im Entwicklungspaket Atmel Studio befindet. Sie müssen somit Atmel Studio installieren um es nutzen zu können.

Dieses kann kostenfrei bei Atmel heruntergeladen werden:

<http://www.atmel.com/tools/ATMELSTUDIO.aspx> bzw. direkt hier:

<http://www.atmel.com/System/BaseForm.aspx?target=tcm:26-64010> (Sie müssen sich registrieren, die Software ist kostenfrei). Achtung: rund 550MB!

Die nachfolgend abgebildeten Bildschirme beziehen sich auf die Version 6; Version 5 ist optisch quasi identisch.

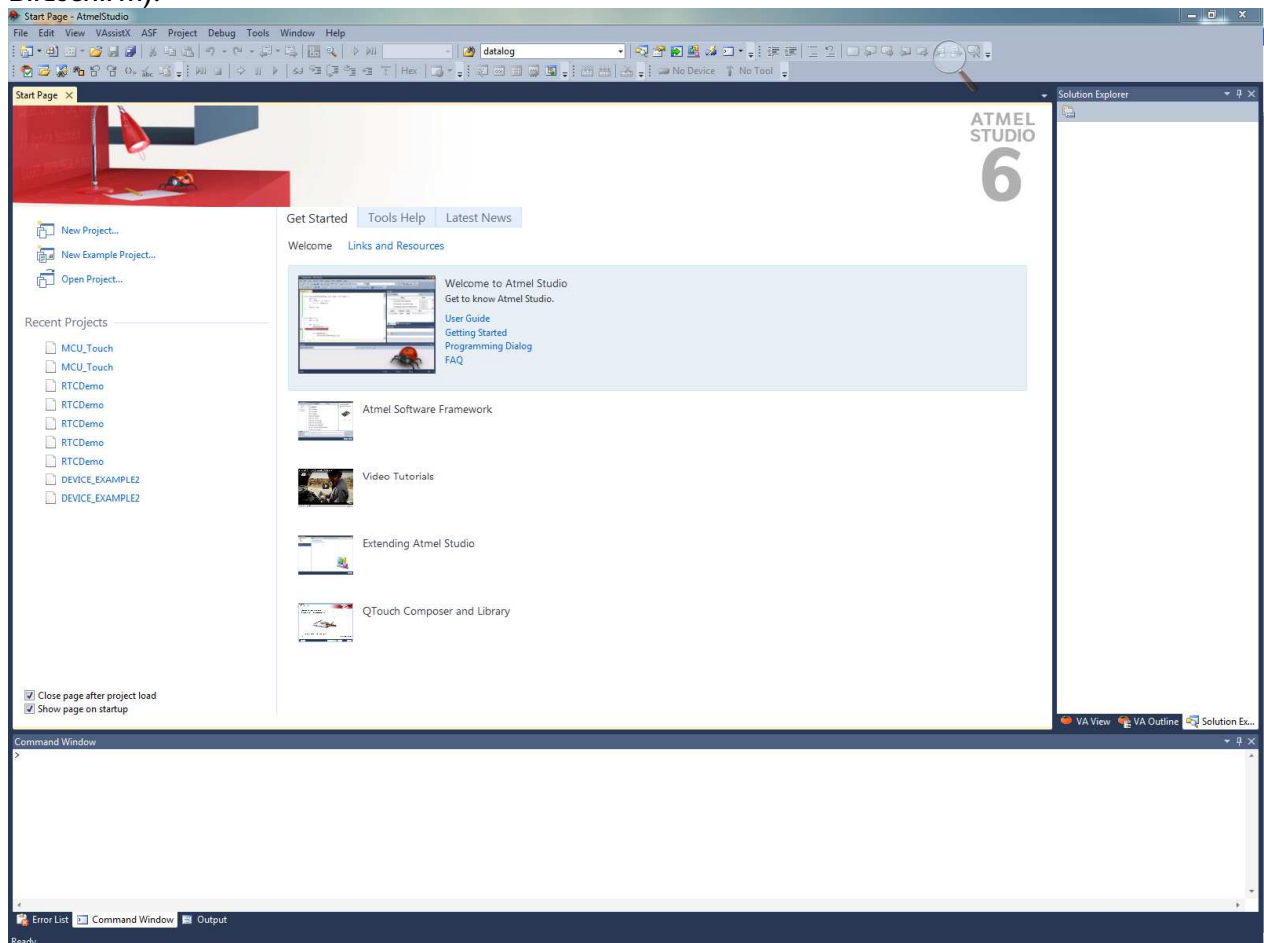
Installation für Windows 7 (vermutlich auch 8)

1. NOCH NICHT den Programmieradapter an USB anschliessen!
wenn schon geschehen siehe gelber Kasten unten
2. Zuerst Atmel Studio installieren
3. Systemsteuerung -> Hardware und Sound -> Gerätemanager
4. rechten Mausklick oben auf den Computernamen -> *Legacyhardware hinzufügen*
5. Beim Willkommensbildschirm auf *Weiter* -> *Hardware manuell aus einer Liste wählen* und installieren
6. "Alle Geräte anzeigen" auswählen -> Weiter
7. Klick auf Datenträger
8. Durchsuchen und folgenden Pfad angeben/suchen:
"C:\Programme\Atmel\AtmelUSB\usb64\windrv6.inf"
bzw. für ein 64bit Windows 7:
"C:\Program Files (x86)\Atmel\AtmelUSB\usb64\windrvr6.inf"
9. OK auswählen
10. WinDriver auswählen -> "Weiter" -> "Weiter" -> "Fertig stellen"
11. Sicherheitsabfrage mit "Installieren" bestätigen -> "Schließen" -> "Schließen"
12. Jetzt müssten unter "Jungo" zwei Geräte erscheinen: WinDriver und AVRISP mkII
13. jetzt erst den USB Programmieradapter an USB anschliessen
14. Fertig – weiter auf der nächsten Seite

Wenn der MKII USB Programmieradapter bereits angeschlossen wurde, ist es sinnvoll, den Treiber erst einmal zu deinstallieren.

- A. Programmieradapter abziehen
- B. Punkte 2 und 3 ausführen (s.o.)
- C. Programmieradapter anschließen
- D. Im Gerätemanager dann mit rechtem Mausklick auf "AVRISP mkII" -> "Deinstallieren" -> OK
- E. weiter bei Punkt 4 (s.o.)

Nach der Installation und dem Starten von Atmel Studio erscheint folgender (oder ähnlicher) Bildschirm):



Wenn Sie nicht innerhalb von AVR Studio entwickeln (z.B. weil Sie mit Bascom oder einem anderen Compiler arbeiten), sondern nur Ihr Programm zum Controller hin übertragen möchten, so brauchen Sie hier nur ein einziges Icon für das Programmierwerkzeug klicken:

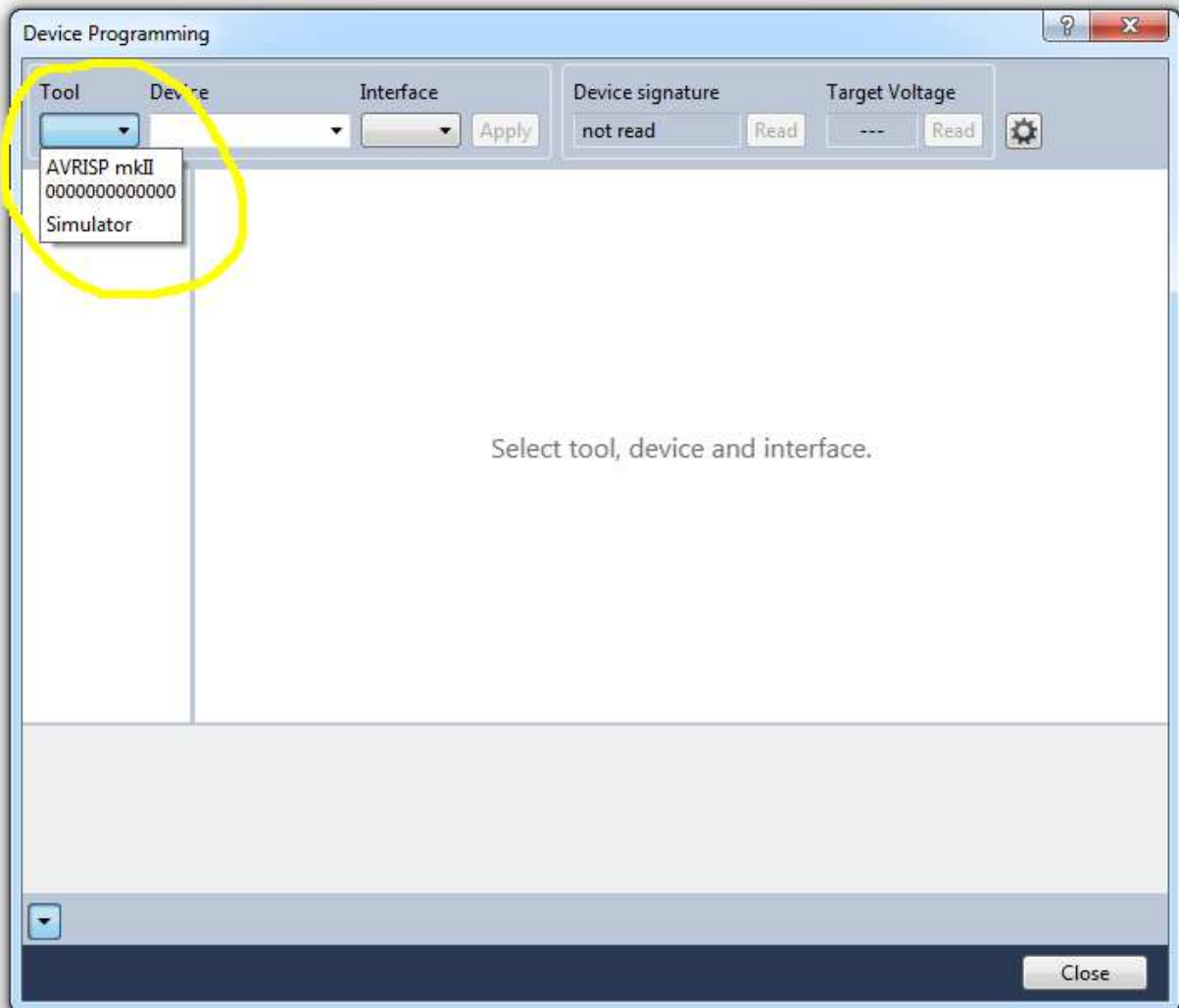


Selbstverständlich nutzen Sie dieses Icon für den Start des Programmiervorgang auch, wenn Sie innerhalb von Atmel Studio entwickeln. Dieses Icon auf der Symbolleiste ist für den Aufruf des Programmierwerkzeugs zuständig.

Der erste Programmiervorgang:

I. Programmieradapter auswählen

Als erstes wählen Sie oben links den Programmieradapter AVRISP mkII aus:



Das Programmierwerkzeug merkt sich diese Auswahl, d.h. zukünftig brauchen Sie hier nur eine erneute Auswahl treffen, wenn sie den Programmieradapter vom USB Bus getrennt haben. In einem der seltenen Fälle, dass sich der Programmieradapter ‚aufgehängt‘ hat, müssen sie ihn einmal von der USB Verbindung trennen und wieder einstecken und dann hier die Auswahl erneut treffen.

2. Auswahl des genutzten Mikrocontrollers

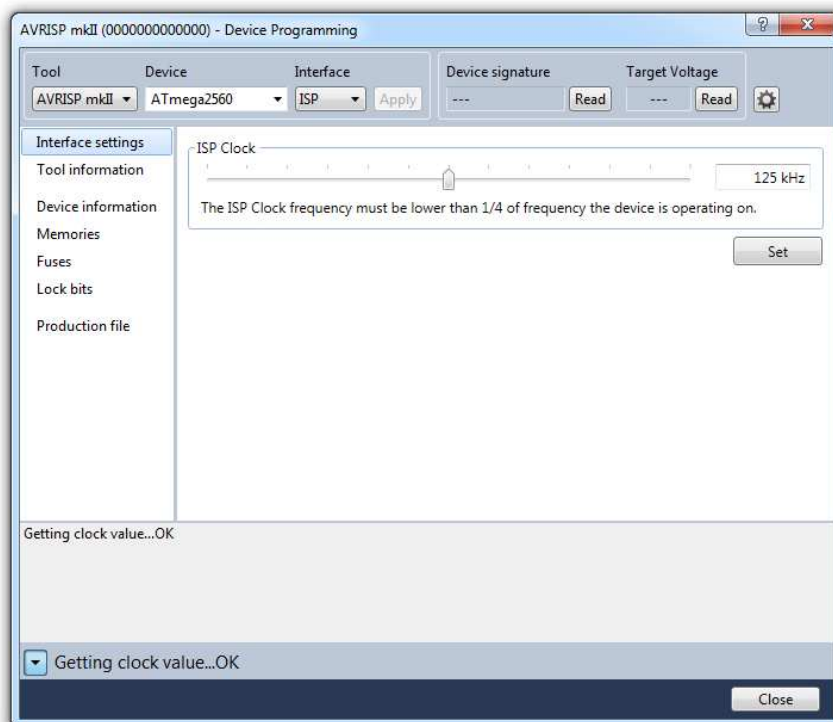
Das Programmierwerkzeug zeigt in der Mitte bereits an „Select tool, device and interface“.

Unter dem Eintrag „Device“ wählen Sie den von Ihnen genutzten Mikrocontroller aus:



Auch diese Einstellung merkt sich das Programmierwerkzeug, d.h. Sie müssen es nur noch bei Änderung des Mikrocontrollers ändern. Das Tool hat für die unterschiedlichen Controllertypen das mögliche Interface bereits abgespeichert. Bei einem ATmega springt es daher automatisch auf „ISP“ und bei einem XMega auf „PDI“.

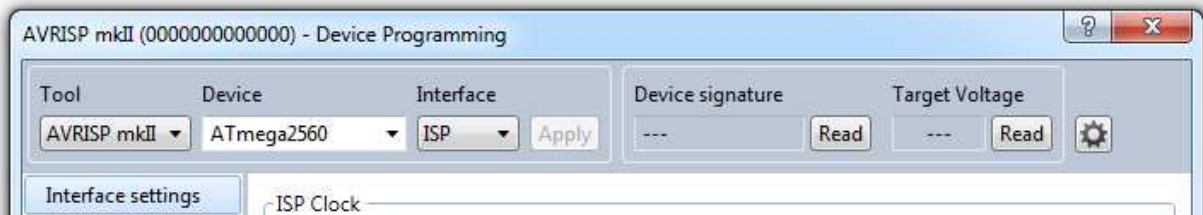
Beim Start des Programmierwerkzeugs und bei Änderungen ist es notwendig, den Button „Apply“ zu drücken. Erst dann wird die Auswahl aktiv und der folgende Bildschirm erscheint:



3. Prüfen der Verbindung

Beim Ersten Anschluss einer Hardware hat es sich als sinnvoll erwiesen, nun erst einmal den Button „Read“ bei „Device Signatur“ zu drücken. Das Programmierwerkzeug prüft nun, ob die ISP/PDI Verbindung stimmt, ob die Einstellungen zur gewählten Hardware passen (z.B. ob der ausgewählte Mikrocontroller zur Hardware passt, ob die anliegende Spannung i.O. ist etc.).

Vorher:



Nach Drücken von „Read“:



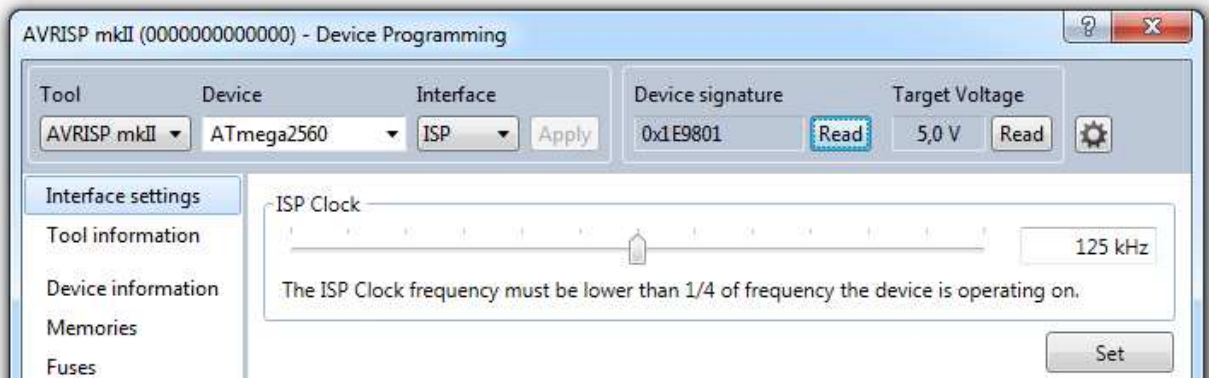
Wenn es hier keine Fehlermeldung gibt, ist alles in Ordnung und einem Programmiervorgang steht nichts im Wege.

Wenn es eine Fehlermeldung gibt, prüfen Sie bitte die Spannungsversorgung, den ISP / PDI Anschluss und ob der Mikrocontroller mit der Auswahl übereinstimmt.

Ein Tipp: Wenn Sie einen nagelneuen Mikrocontroller oder ein neu erworbenes Mikrocontrollerboard erwerben, dann setzen Sie unbedingt die ISP Programmierfrequenz (siehe nächster Absatz) auf 125 KHz zurück. Sonst können Sie lange nach dem Fehler suchen (Erläuterung s.u.)

4. Auswahl der ISP Programmierfrequenz

Unter dem linken Reiter „Interface settings“ wählen Sie die Programmiergeschwindigkeit des Programmierertools aus (bei ISP Programmierung; XMegas mit PDI Programmierung benötigen dies nicht). Der Text unterhalb des Sliders besagt, dass die Programmier-Clock-Frequenz niedriger als ein Viertel des Mikrocontrollers-Systemtakts sein muss.

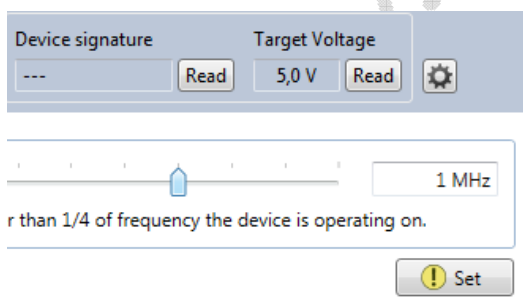


Daher gilt:

- Systemtakt 1 Mhz: max. Programmierfrequenz 125 Khz
- Systemtakt 4 Mhz: max. Programmierfrequenz 500 Khz
- Systemtakt 8 Mhz: max. Programmierfrequenz 1 Mhz
- Systemtakt 14,7456 Mhz: max. Programmierfrequenz 2 Mhz
- Systemtakt 16 Mhz: max. Programmierfrequenz 2 Mhz
- Systemtakt 20 Mhz: max. Programmierfrequenz 4 Mhz

Neue Atmel-Mikrocontrollerchips sind i.d.R. auf 1 Mhz vorprogrammiert, d.h. die maximale Frequenz muss bei der Erstprogrammierung unterhalb von 250 Khz liegen – bei diesem Programmieradapter also 125 Khz.

Erst nach dem Ändern der Fuse-Bits (s.u.) für eine höhere Mikrocontroller-Systemtaktfrequenz, können Sie die Taktfrequenz des Programmieradapters hochsetzen.



Vergessen Sie nicht, nach dem Ändern der Frequenz, den Button „Set“ zu drücken (er zeigt dann ein Ausrufezeichen als Erinnerung). Solange dieser nicht gedrückt wird, wird die Änderung nicht aktiv.

Ein Drücken von „Read“ unter „Device Signature“ zeigt, ob zum Mikrocontroller eine ordnungsgemäße Verbindung aufgebaut werden kann.

Anmerkungen:

- 1) Unsere Mikrocontrollerboards sind üblicherweise immer mit einem Demo-/Testprogramm vorprogrammiert und wir setzen die Fuses direkt korrekt, d.h. bei unseren Boards können Sie direkt 1 Mhz als ISP-Frequenz einstellen. Andere Anbieter bieten diesen Service i.d.R. nicht, hier müssen Sie erst 125 Khz einstellen.
- 2) Widerstehen Sie der Versuchung, mit einem Clocktakt von genau $\frac{1}{4}$ des Systemtakts zu arbeiten (also z.B. 4 Mhz bei einem 16 Mhz Controller). Auch wenn es erst so aussieht als ob das ebenfalls funktioniert: sie erreichen dadurch fehlerhaft beschriebene Speicherstellen.

Die folgende Tabelle zeigt die Dauer des Programmiervorgangs bei einem Programm, welches einen 256 KByte Flash zu 100% ausnutzt. Die Zeiten zeigen die Dauer des Programmiervorgangs ohne Verify. Programme mit weniger Speicherbedarf sind entsprechend schneller.

Programmierfrequenz	256 KByte Flash schreiben	256 Kbyte Flash lesen
Serieller Programmieradapter (AVR910/911 etc.)	650 sek. (oder noch langsamer)	265 sek.
„Billiger“ USB Adapter	ca. 90 sek.	ca. 80 sek.
Atmel mk2; Diamex ALLAVR mit 125 khz	86 sek.	81 sek.
Atmel mk2; Diamex ALLAVR mit 1 Mhz	16 sek	11 sek
Atmel mk2; Diamex ALLAVR mit 2 Mhz	12 sek.	6 sek

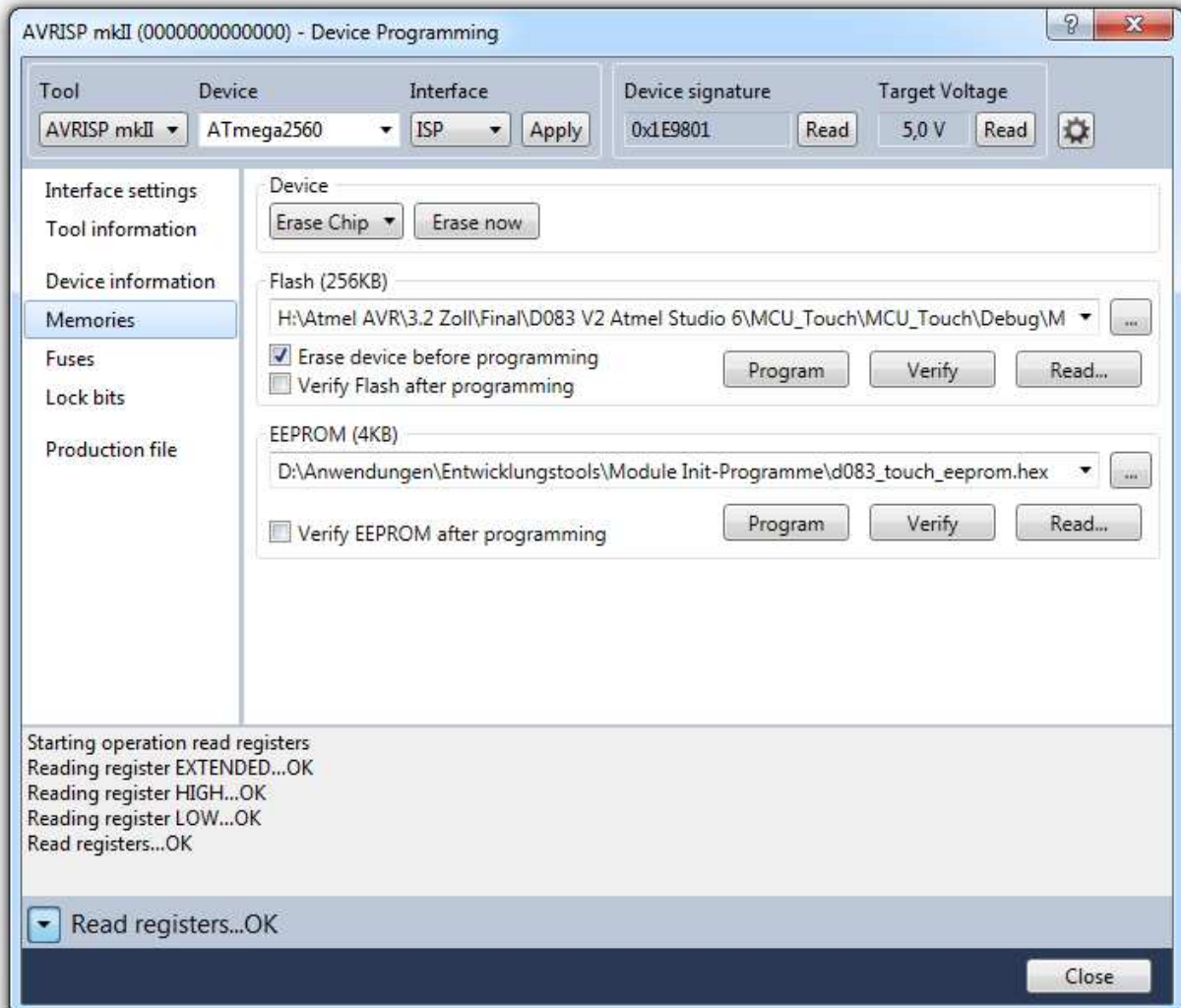
Mit Verify: addieren der Zeiten von Schreiben und Lesen

Während der Entwicklungsphase wird ein Programm u.U. mehrere hundert Male zum Test in den Chip programmiert. Sie können sich also vorstellen, wie viele Stunden Sie mit einer langsamen Einstellung bzw. einem langsamen Programmieradapter verlieren.


Das Tool merkt sich die Auswahl der ISP Programmierfrequenz. Sie brauchen diese Auswahl also nur 1x zu treffen. Ausnahme: wie bereits erwähnt, bei einem neuen, unprogrammierten Chip müssen Sie zuerst die Programmierfrequenz auf 125 Khz zurücksetzen.

5. Programmiervorgang

Alle oben beschriebenen Einstellungen müssen Sie – solange Sie nicht das Board / den Mikrocontroller wechseln – nur 1x vornehmen. Das Programmiertool merkt sich die Einstellungen, d.h. Sie können nach dem Start immer direkt den Reiter „Memory“ wählen.



Die Betätigung des Button „Erase Chip“ ist normalerweise nicht notwendig, denn der Chip wird sowieso während des Programmiervorgangs gelöscht.

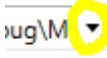
Unter Flash wählen Sie durch Klick auf den Button  die zu programmierende Datei. In der Regel lautet der Dateiname „Programmname.hex“, d.h. Sie wählen die vom Compiler erstellte .hex Datei aus. Achten Sie darauf dass „Erase device before programming“ ausgewählt ist, ansonsten wird der Programmiervorgang fehlschlagen. Die Option „Verify flash after programming“ zur Überprüfung der gerade programmierten Daten können Sie bei der finalen Programmierung zur Sicherheit nutzen; während der Entwicklungsphase ist das nur Zeitverschwendung.

Durch Druck auf den Button „Programm“ im Bereich „Flash“ wird der Programmiervorgang gestartet. Sofern die Option „Verify flash...“ angeklickt wurde, wird nach dem Programmiervorgang der Flash-Speicher auf korrekte Programmierung geprüft. Ansonsten lässt sich dieser Vorgang auch immer manuell mit dem Button „Verify“ starten.

Der Button „Read“ eignet sich zum Auslesen des Programmspeichers. So können Sie z.B. das Test- und Demoprogramm auf unseren Mikrocontrollerboards auslesen und als .hex-Datei abspeichern, so dass Sie dieses jederzeit wieder einspielen können.

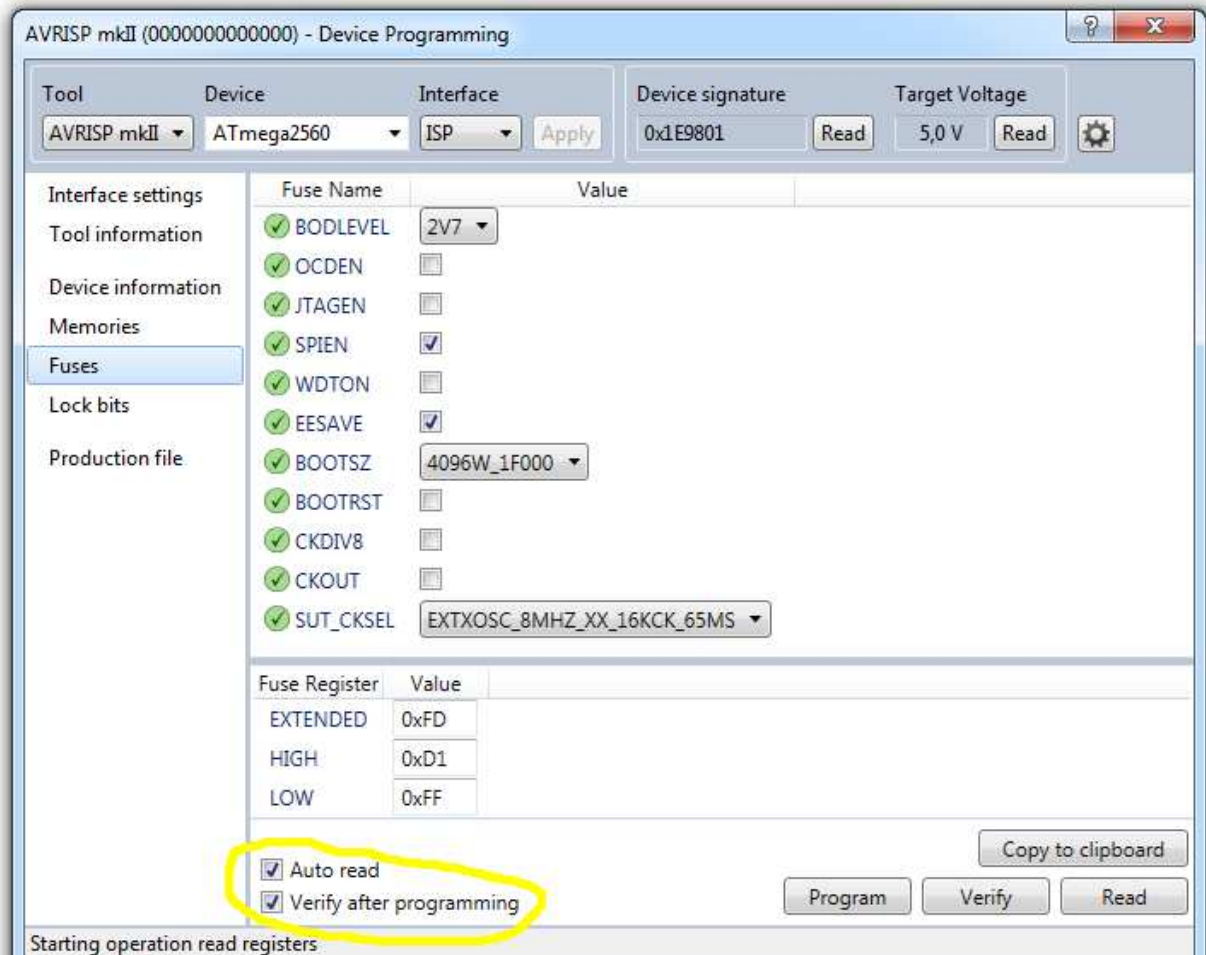
Sofern Sie Voreinstellungsdaten in das Eeprom des Mikrocontrollers schreiben möchten, führen Sie die obige Prozedur auch mit der Eeprom Funktion des Programmiertools durch. Ansonsten brauchen Sie kein Eeprom zu beschreiben.

Das Programmiertool merkt sich die zuletzt programmierte .hex-Datei, d.h. diese Auswahl brauchen Sie während der Entwicklung nicht laufend durchführen.

Tipp: Unter dem kleinen schwarzen Dreieck  können Sie zwischen den fünf zuletzt aufgerufenen .hex-Dateien wählen. Das ist sehr praktisch, wenn Sie zeitgleich an verschiedenen Projekten arbeiten.

6. Fuses / Fuse-Bits

Fuses sind quasi Voreinstellungen des Mikrocontrollers. Hier legen Sie u.a. die Taktfrequenz/-quelle und andere Parameter fest.



Ganz wichtig ist es, dass die beiden gelb markierten Auswahloptionen immer eingeschaltet sind. Ohne „Auto Read“ prüft das Tool ansonsten niemals selbsttätig, ob der verbundene Mikrocontroller zur aktuellen Auswahl passt. Zudem erleichtert es das Ändern, denn das vorherige Einlesen und Darstellung der aktuellen Fuses hilft bei der Vermeidung von Falsch-einstellungen.

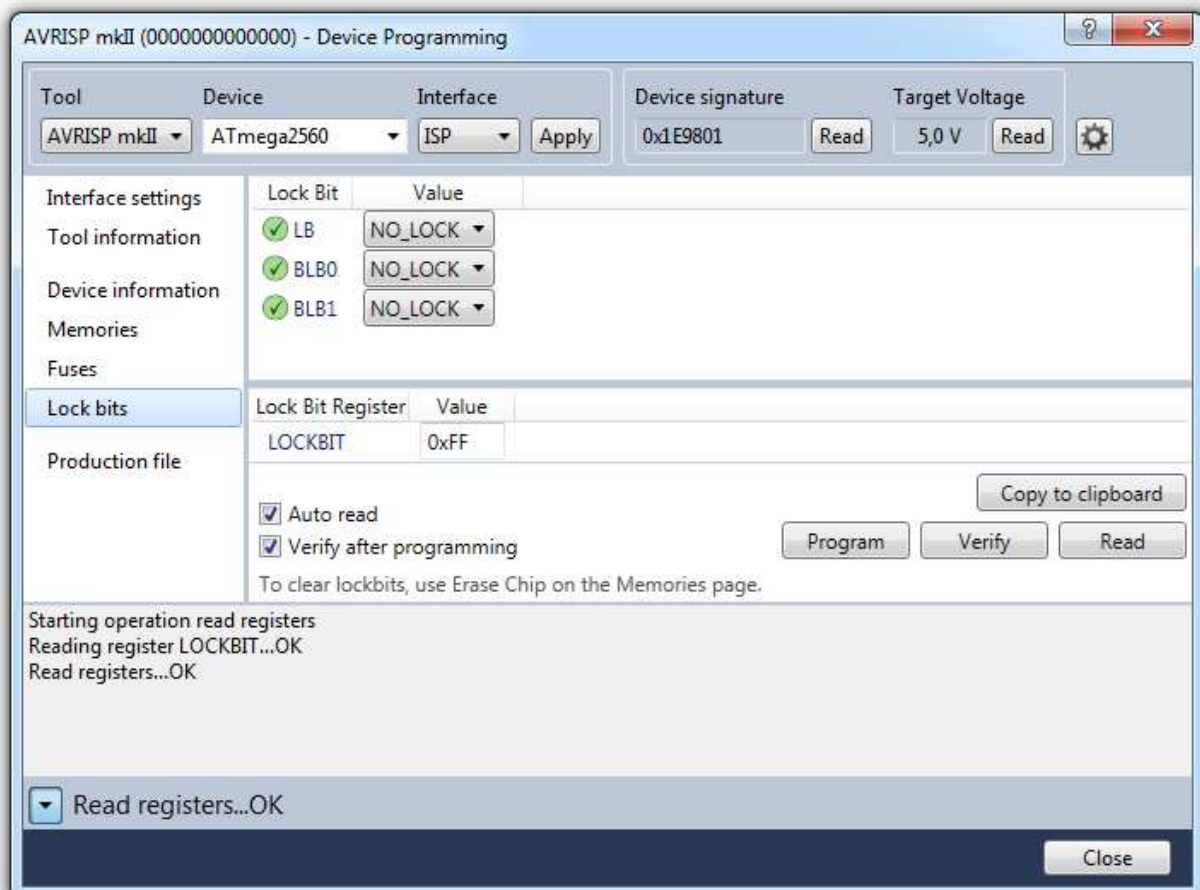
Vorsicht: Das falsche Setzen von Fuse-Bits z.B. kann fatal sein. Durch eine falsche Programmierung der Fuse-Bits können Sie die Voreinstellungen des Mikrocontrollers so ändern, dass Sie keinen Zugriff mehr haben – auf Deutsch: sie haben Elektroschrott produziert.

Wenn Sie sich nicht damit auskennen: Lassen Sie im Zweifel die Finger weg! Da jeder Mikrocontroller unterschiedliche Fusebits hat, müssen Sie sich hier selber zusammen mit dem Datenblatt des Controllers durchbeißen und sich evtl. auch 1 oder 2 Controller „zerschießen“. Das gehört wohl zum Lernprozess dazu.... so wie das Verbrennen an der heißen Herdplatte als Kind.

Im Zweifel rufen Sie uns einfach an, bevor Sie ein Fusebit ändern.

7. Lock Bits

Mit den Lockbits können Sie z.B. verhindern, dass jemand einen programmierten Controller wieder ausliest und die Hardware inkl. dieses Programms vervielfältigt. Besonders bei kommerziellen Projekten sind die Lockbits i.d.R. gesetzt.



Ansonsten gilt hier ähnliches wie bei den Fuse-Bits: Die Lock-Bits sind je nach Controller unterschiedlich. Konsultieren Sie das entsprechende Kapitel im Datenblatt des Mikrocontrollers.

Sonstige Hilfe und Informationen

Innerhalb des Programmier-Tools erhalten Sie durch Drücken von F1 in jeder Rubrik der Software eine umfangreiche (englische) Hilfe.

So, dass soll für diese Kurzeinführung reichen.

Wir wünschen Ihnen viel Erfolg!

Anhang: AVR MKII / ALLAVR und Bascom

Immer wieder wird versucht, den AVR MKII direkt aus Bascom heraus anzusprechen. Offen gesagt: wir haben es aufgegeben. Alle ermittelten Lösungen sind entweder langsam oder unzuverlässig (d.h. bei dem einen User/Rechner klappt es, bei dem anderen nicht), sodass wir dieses Vorgehen nicht empfehlen können. Wir supporten es auch nicht, also sparen Sie sich bitte die Anfragen.

Unsere Empfehlung ist:

- Programmieren Sie in Bascom und kompilieren Sie wie üblich (Taste F7).
- Bascom erzeugt nun eine Datei *programmname.HEX*
- Wählen Sie diese .HEX Datei nun aus der Programmiersoftware des AVR Studio heraus aus und senden sie an den Mikrocontroller.
- Lassen Sie das Atmel Studio während der Entwicklung einfach im Hintergrund offen und schalten dann mit den Tasten ALT-TAB einfach zwischen Bascom und Atmel Studio hin und her.

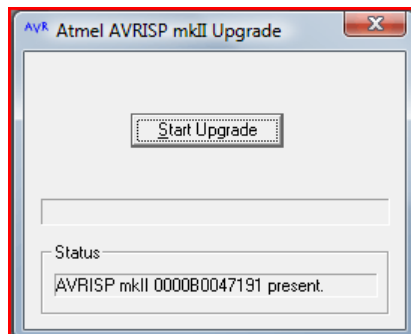
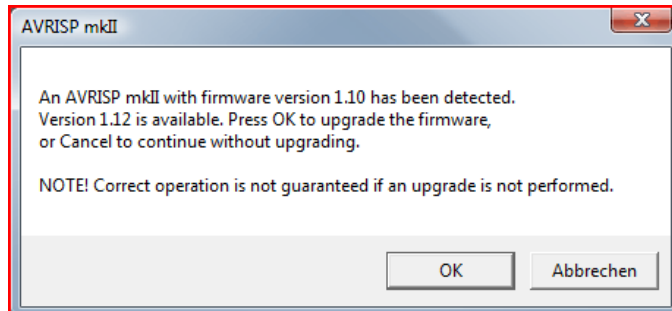
Somit ist der Aufwand während der Entwicklung (bis auf das erstmalige Starten von Atmel Studio) quasi der gleiche. Dafür sparen Sie aber bei jedem Programmiervorgang viele Sekunden Wartezeit ein, die sich am Ende des Tages durchaus auf 30 Minuten Zeitersparnis summieren können.

Vorteile:

- Schnelle Durchführung des Programmiervorgangs: Was in Bascom evtl. 2 Minuten braucht, ist mit dem AVR Studio (dank hochgestellter ISP Programmierfrequenz) in 10 Sekunden erledigt
- Der Programmiervorgang blockiert Bascom nicht: Während des Programmiervorgangs können Sie in Bascom bereits weiterarbeiten

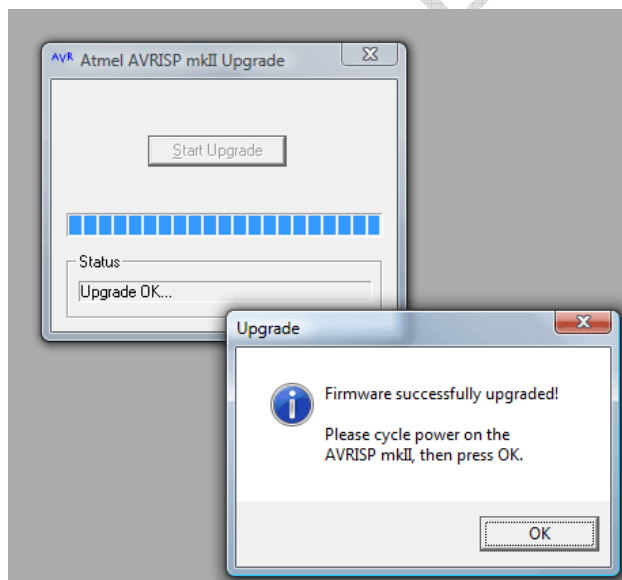
Anhang: Firmware update

Beim allerersten Start erkennt AVR Studio u.U., dass es eine neuere Firmware für den Programmier gibt. Wenn dem so ist, sollten Sie diese nun installieren lassen. Drücken Sie „OK“



Das Upgrade startet nun und ist nach wenigen Sekunden beendet.

Wichtig ist nun, dass Sie nach dem Upgrade den Programmieradapter einmal kurz von der USB Schnittstelle abziehen und wieder anstecken.

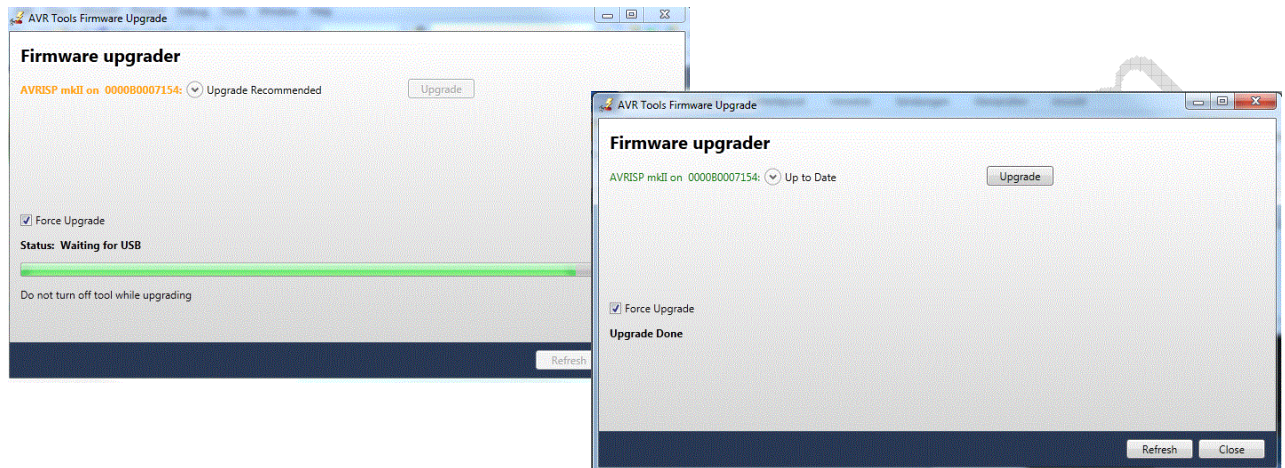


Anhang: Unterschiede bei Atmel Studio 5 / 6

Screenshots

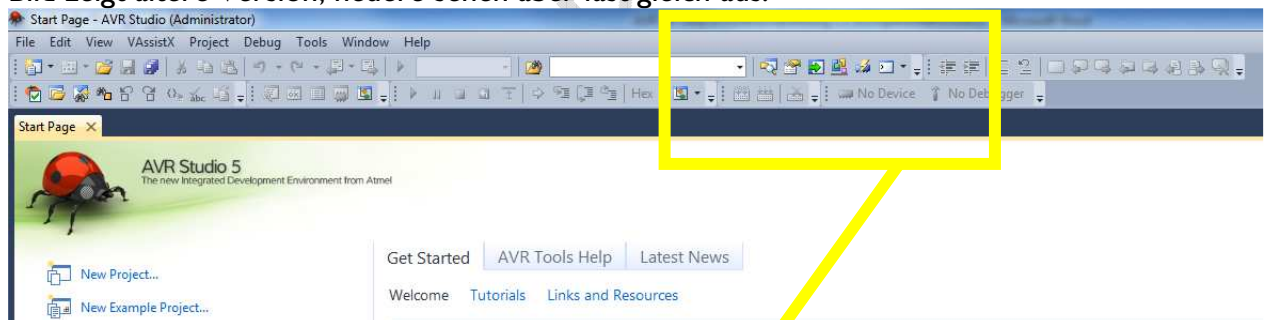
Einige Bildschirme der Version 5 und 6 sehen etwas anders als in der Version 4 aus. Zur Erleichterung bilden wir hier die wichtigsten ab:

Firmware Upgrade des Programmieradapters



Menü / Icon-Leiste Startbildschirm vom AVR Studio

Bild zeigt ältere Version, neuere sehen aber fast gleich aus.



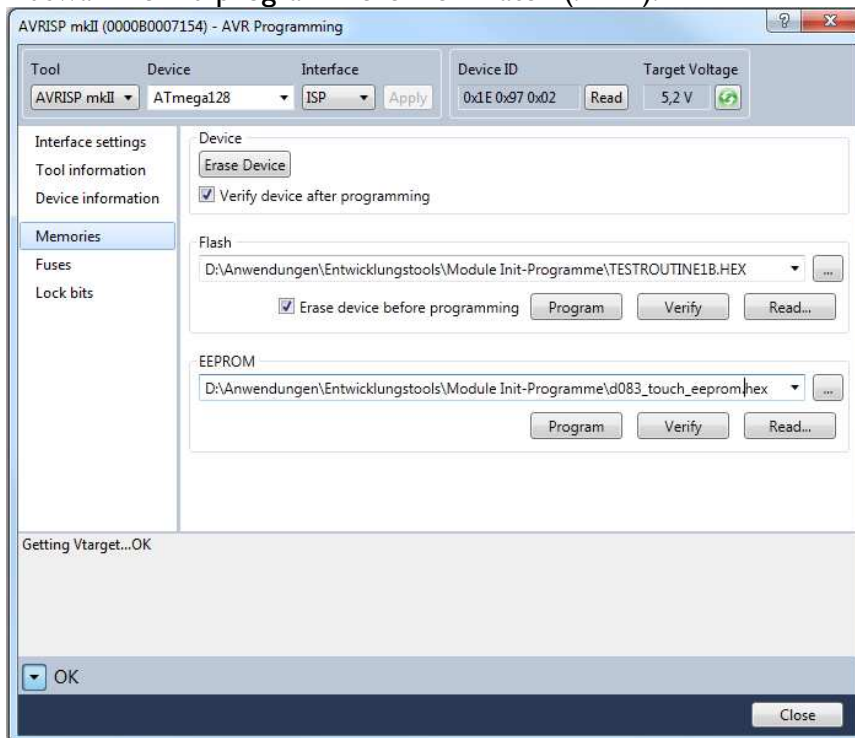
Icon zum Start der Programmiersoftware:



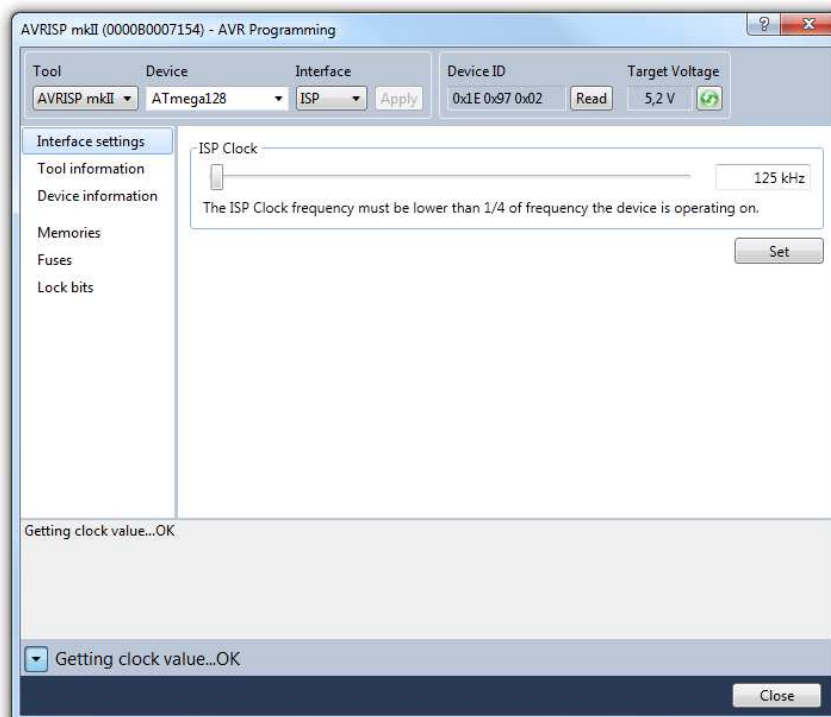
Der Knopf mit dem „Blitz“ startet die Programmiersoftware
(Alternativ im Menü: „Tools – AVR Programming“)

Das Programmier-Tool

Auswahl der zu programmierenden Daten (.HEX):



Festlegung der Geschwindigkeit des Programmier-Tools:



Neue Chips: 125 Khz

Chips mit externem Quarz und angepasster Fuse für Taktfrequenz: 1 Mhz

Typische Probleme:

Ich kann keine Verbindung zum Programmer aufbauen.

Lösung 1: Kontrollieren, ob die LED innen am USB Eingang des Programmieradapters leuchtet. Wenn nein: Programmieradapter kurz abziehen und wieder anstecken.

Lösung 2: Wenn auch nach einem Herunterfahren (inkl. Ausschalten) des Rechners das Problem immer noch besteht (dies sollte eigentlich generell immer die erste Aktion sein, bevor man irgendwo an der Softwareinstallation herumspielt – Merke: „Boot tut gut“), hilft in der Regel nur die Deinstallation und Neuinstallation des USB Treibers.

Die Leuchtdiode im Innern des Programmers schaltet immer mal wieder ab und ich muss den Programmer vom USB Port trennen und wiederverbinden.

Lösung: Der Programmieradapter ist empfindlich gegen statische Aufladung und stürzt dann schon mal ab, wenn man an seiner Zielschaltung arbeitet oder den Programmieradapter aufsteckt. Dagegen gibt es kein Patentrezept. Grundsätzlich sollten Sie eigentlich bei der Arbeit mit empfindlicher Elektronik durch ein Armband geerdet sein, so dass das Problem dann nicht mehr auftritt.

Ich kann keine Verbindung zu meinem Controller aufnehmen. Beim Drücken von „Read“ bei „Device Signature“ erscheint nur ----

Lösung: Evtl. ist die ISP Frequenz zu hoch eingestellt. Bitte setzen Sie diese auf 125 khz, notfalls auch testweise einmal auf eine geringe Frequenz.

Beim Programmieren erhalte ich immer mal wieder eine Dialogbox mit Fehlern, nach ein paar Mal Drücken auf „OK“ zum Wiederholen geht es dann.

Das Problem tritt i.d.R. bei einer nicht ordnungsgemäßen Stromversorgung des Controllerboards auf (z.B. Netzteil mit zu wenig Leistung).

Ich habe in den Fuses etwas verstellt und nun bekomme ich keinen Kontakt mehr zum Controller.

Lösung: Das ist Pech, da haben Sie wohl versehentlich eine falsche Fuse erwischt. Sie können versuchen, an XTALI ein Taktsignal anzulegen (Funktionsgenerator oder ein zweiter Mikrocontroller) und dann versuchen, die Fuses auszulesen und zu ändern. Lesen Sie hierzu auch die Informationen auf Seite 8).

Wenn dies nicht klappt, ist der Controller vermutlich reif für den Elektroschrott (sofern Sie keinen Hochvoltprogrammer wie z.B. ein STK 600 mit passendem Aufsatz besitzen – aber dazu müssen Sie vorher den Controller von der Platine ablöten).

Ich habe keine Fuses verstellt, trotzdem bekomme ich keinen Kontakt mehr zum Mikrocontroller.

Lösung: Hier gibt es viele Möglichkeiten (meist ist durch statische Aufladung der Chip zerstört worden). Unser Tipp: Immer mit Erdungsarmband an der Elektronik arbeiten. Hier hilft nun nur selber nach der Fehlerquelle suchen und notfalls ist der Austausch des Mikrocontrollers notwendig. Lesen Sie hierzu auch die Informationen auf Seite 8 und probieren Sie sie aus, oftmals werden Fuses auch unabsichtlich verstellt.

Die LED leuchtet, ich kann aber immer noch nicht Kontakt zum Programmer aufnehmen oder ich bekomme keinen Kontakt mehr zum Controller.

Lösung: Ein paar wenige Male ist es vorgekommen, das Atmel Studio nicht mehr korrekt gearbeitet hat und Windows neu gestartet werden musste. Bei unerklärlichen Problemen bietet sich dies an, bevor man vergeblich stundenlang einen gar nicht vorhandenen Fehler sucht.

Ich habe mir ein eigenes Board gebaut und ich kann es nicht mit dem Programmieradapter ansprechen.

Lösung: Vielleicht haben Sie einen typischen Fehler begangen, und haben die MOSI / MISO-Leitungen vom ISP Programmierstecker einfach mit den MOSI / MISO-Leitungen des Mikrocontrollers verbunden? Bei manchen Controllern (z.B. ATMega128) dürfen die Programmierleitungen NICHT an MOSI / MISO angeschlossen werden (dort ist es z.B. Rx0/Tx0). Lesen Sie dazu das Datenblatt des Controllers.

Kontakt:

Speed IT up
Inhaber Peter Küsters
Wekeln 39
47877 Willich
Telefon: (0 21 54) 88 27 5-10
Telefax: (0 21 54) 88 27 5-22

Weitere Informationen und Updates: www.display3000.com

Autor dieses Manuals: Peter Küsters.
© aller Informationen: Peter Küsters

© www.Display3000.com