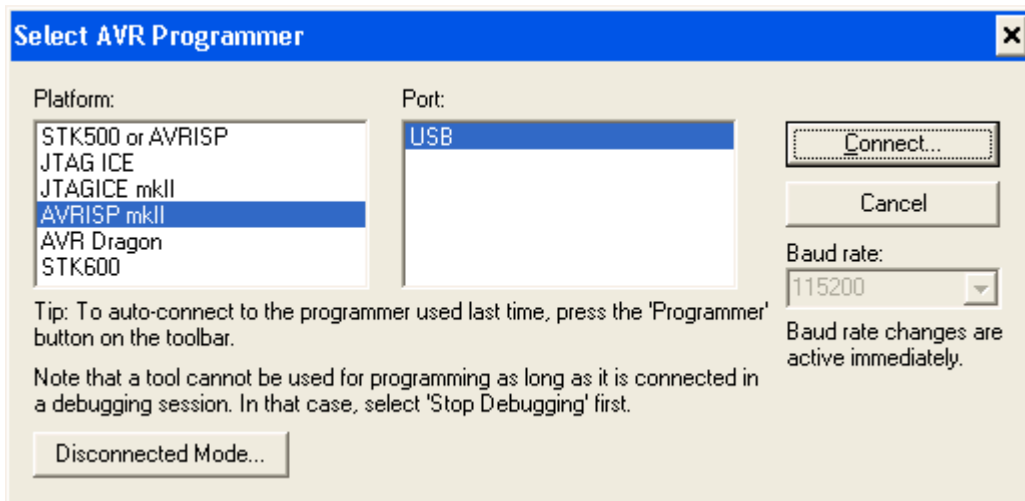


Programmierung BMC Mikro

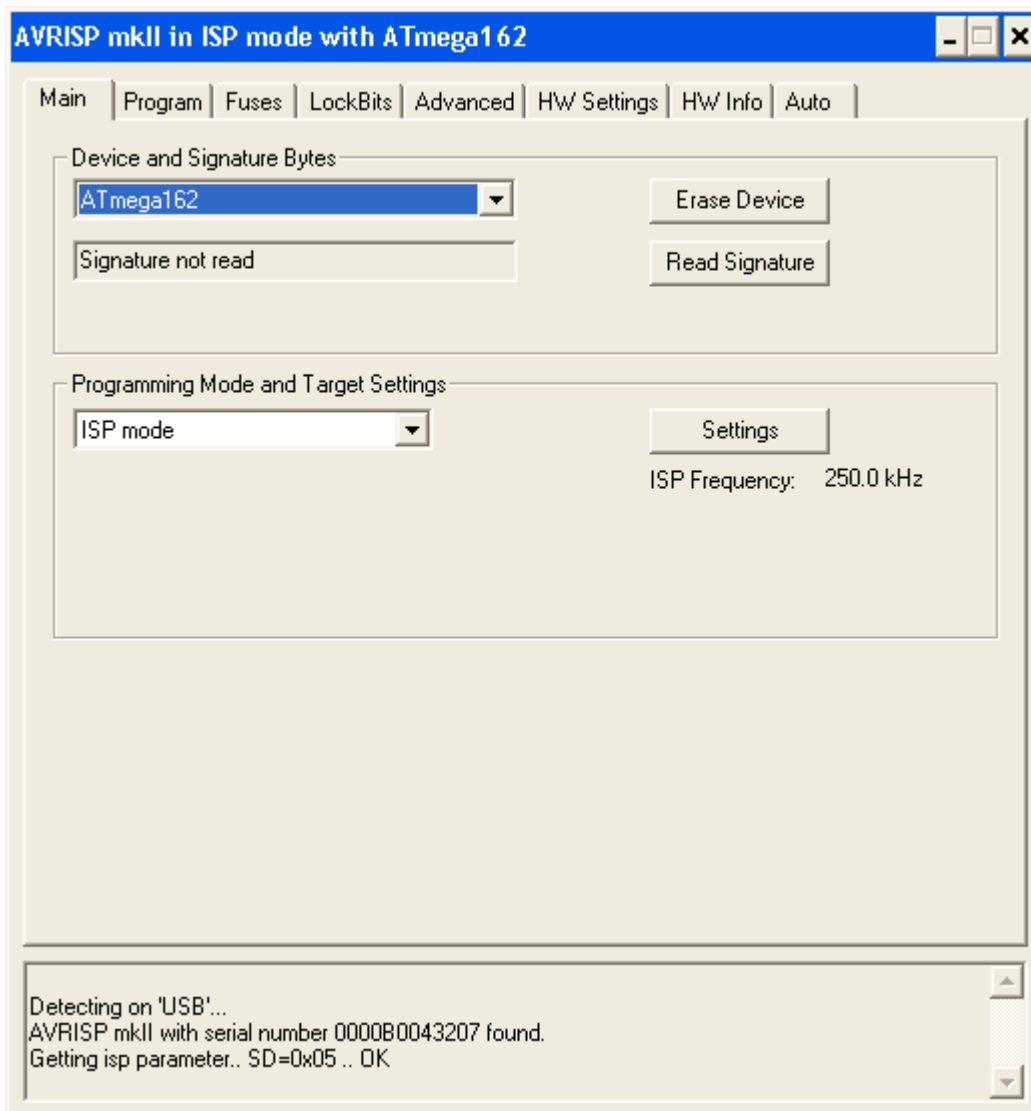
AVR Studio 4 öffnen

Es ist nicht notwendig, ein Projekt zu öffnen (nur für Assembler Files)

Nur aus „Tools / Program AVR / Connect“ den richtigen Programmer wählen



➔ Später kann direkt über den schwarzen „AVR“ Button das Programmer Menü geöffnet werden.



Den richtigen Controller angeben, die ISP Frequenz checken.

Eine Frequenz von 1 MHz führte zu keinen brauchbaren Ergebnissen bei den Fuse Bits.

250 KHz funktionierten im ersten Test einwandfrei.

Später mit 500 KHz probiert, geht auch nicht. Somit ist 250 KHz die höchste einsetzbare Frequenz.

Angabe der hex- und eep-Dateien

The screenshot shows the 'AVRISP mkII in ISP mode with ATmega162' window. It has a tabbed interface with 'Main', 'Program', 'Fuses', 'LockBits', 'Advanced', 'HW Settings', 'HW Info', and 'Auto'. The 'Main' tab is active. It contains sections for 'Device', 'Flash', 'EEPROM', and 'ELF Production File Format'. The 'Device' section has an 'Erase Device' button and two checked options: 'Erase device before flash programming' and 'Verify device after programming'. The 'Flash' section has radio buttons for 'Use Current Simulator/Emulator FLASH Memory' (unselected) and 'Input HEX File' (selected). The 'Input HEX File' text box contains 'D:\MoBaSbS\AVR Studio\BMC.hex'. Below it are 'Program', 'Verify', and 'Read' buttons. The 'EEPROM' section has similar radio buttons and a selected 'Input HEX File' text box containing 'D:\MoBaSbS\AVR Studio\BMC.eep', with 'Program', 'Verify', and 'Read' buttons below. The 'ELF Production File Format' section has an 'Input ELF File' text box and 'Program' and 'Save' buttons. A note states: 'Fuses and lockbits settings must be specified before saving to ELF'. At the bottom, a status area shows: 'Detecting on 'USB'...', 'AVRISP mkII with serial number 0000B0043207 found.', and 'Getting isp parameter.. SD=0x05 .. OK'.

AVRISP mkII in ISP mode with ATmega162

Main Program Fuses LockBits Advanced HW Settings HW Info Auto

Device

Erase Device

☒ Erase device before flash programming ☒ Verify device after programming

Flash

☐ Use Current Simulator/Emulator FLASH Memory

☒ Input HEX File D:\MoBaSbS\AVR Studio\BMC.hex ...

Program Verify Read

EEPROM

☐ Use Current Simulator/Emulator EEPROM Memory

☒ Input HEX File D:\MoBaSbS\AVR Studio\BMC.eep ...

Program Verify Read

ELF Production File Format

Input ELF File ...

Program Save

Fuses and lockbits settings must be specified before saving to ELF

Detecting on 'USB'...

AVRISP mkII with serial number 0000B0043207 found.

Getting isp parameter.. SD=0x05 .. OK

Fuse Bits ATmega 162

AVRISP mkII in ISP mode with ATmega162

Main

Program

Fuses

LockBits

Advanced

HW Settings

HW Info

Auto

M161C	<input type="checkbox"/>
BODLEVEL	Brown-out detection at VCC=4.3 V
OCDEN	<input type="checkbox"/>
JTAGEN	<input type="checkbox"/>
SPIEN	<input checked="" type="checkbox"/>
WDTON	<input type="checkbox"/>
EESAVE	<input checked="" type="checkbox"/>
BOOTSZ	Boot Flash size=1024 words start address=\$1C00
BOOTRST	<input type="checkbox"/>
CKDIV8	<input type="checkbox"/>
CKOUT	<input type="checkbox"/>
SUT_CKSEL	Ext. Crystal Osc. 8.0- MHz; Start-up time: 16K CK + 65 ms

EXTENDED	0xF9
HIGH	0xD1
LOW	0xFF

☒ Auto read

☒ Smart warnings

☒ Verify after programming

Program

Verify

Read

Setting mode and device parameters.. OK!

Entering programming mode.. OK!

Reading fuses address 0 to 2.. 0xFF, 0xD1, 0xF9 .. OK!

Leaving programming mode.. OK!

ATTiny 2313 Fuse Bits

The screenshot shows the 'Fuses' tab of the AVRISP mkII in ISP mode with ATTiny2313. The window has a blue title bar and a menu bar with 'Main', 'Program', 'Fuses', 'LockBits', 'Advanced', 'HW Settings', 'HW Info', and 'Auto'. The 'Fuses' tab is active, displaying a list of fuse bits with checkboxes and dropdown menus. Below this list is a section for 'EXTENDED' fuses with three rows: 'HIGH' and 'LOW' both set to '0xFF'. At the bottom of the tab are three checkboxes: 'Auto read' (checked), 'Smart warnings' (checked), and 'Verify after programming' (checked). To the right of these checkboxes are three buttons: 'Program', 'Verify', and 'Read'. At the very bottom of the window is a text area showing the status of the programming process.

Fuse Bit	Value
SELFPRGEN	<input type="checkbox"/>
DWEN	<input type="checkbox"/>
EESAVE	<input checked="" type="checkbox"/>
SPIEN	<input checked="" type="checkbox"/>
WDTON	<input type="checkbox"/>
BODLEVEL	Brown-out detection at VCC=4.3 V
RSTDISBL	<input type="checkbox"/>
CKDIV8	<input type="checkbox"/>
CKOUT	<input type="checkbox"/>
SUT_CKSEL	Ext. Crystal Osc. 8.0- MHz; Start-up time: 14 CK + 65 ms

Fuse Bit	Value
EXTENDED	0xFF
HIGH	0x99
LOW	0xFF

☒ Auto read
☒ Smart warnings
☒ Verify after programming

Program Verify Read

Entering programming mode.. OK!
Writing fuses address 0 to 2.. 0xFF, 0x99, 0xFF .. OK!
Reading fuses address 0 to 2.. 0xFF, 0x99, 0xFF .. OK!
Fuse bits verification.. OK
Leaving programming mode.. OK!

Programmiert über das ISP Board von Patrick

Zuerst die Signatur des μ C lesen, dann die Fuse Bits setzen und schreiben, noch mal Lesekontrolle.
Immer Flash und EEPROM schreiben.

Für einen von drei ATTinys musste die Taktfrequenz am AVR MK II von 250 KHz auf 125 KHz reduziert werden.

Fuse Bits ATmega 8

The screenshot shows the 'Fuses' tab of the AVRISP mkII software. The window title is 'AVRISP mkII in ISP mode with ATmega8'. The tabs are: Main, Program, Fuses, LockBits, Advanced, H/W Settings, H/W Info, and Auto. The Fuses tab contains a list of fuse bits with checkboxes and dropdown menus. Below the list are checkboxes for 'Auto read', 'Smart warnings', and 'Verify after programming', followed by 'Program', 'Verify', and 'Read' buttons. At the bottom is a status window showing progress messages.

Fuse Bit	Value / Status
RSTDISBL	<input type="checkbox"/>
WTDON	<input type="checkbox"/>
SPIEN	<input checked="" type="checkbox"/>
EESAVE	<input checked="" type="checkbox"/>
BOOTSZ	Boot Flash size=1024 words Boot address=\$0C00
BOOTRST	<input type="checkbox"/>
CKOPT	<input type="checkbox"/>
BODLEVEL	Brown-out detection at VCC=4.0 V
BODEN	<input checked="" type="checkbox"/>
SUT_CKSEL	Ext. Crystal/Resonator High Freq.; Start-up time: 16K CK + 64 ms

Mode	Value
HIGH	0xD1
LOW	0x3F

☒ Auto read
☒ Smart warnings
☒ Verify after programming

Program Verify Read

Setting mode and device parameters.. OK!
Entering programming mode.. OK!
Reading fuses address 0 to 1.. 0xE1, 0xD9 .. OK!
Leaving programming mode.. OK!

Inbetriebnahme

TCL Interpreter installiert (empfohlene Version beachten, 2009-09-05: 8.4.13.0.261555

Aktuelle MoBaSbS-Konfig-Software geladen

Achtung:

bei Updates der Konfig-Software diese immer in ein leeres Verzeichnis installieren, eventuell die .INI-Datei übernehmen

Bedienung, Updates usw. siehe MoBaSbS V4 Handbuch bzw. PDF zur Konfig-Software

COM Port entsprechend eingestellt, BM-Report aktualisieren und übernehmen.

Dabei wird die ausgelesene HW-Info in den Speicher der Konfig-Software übernommen und kann als Datei gespeichert werden. Wird der vorgegebene Name verwendet, kann dieser auch beim Start der Software gleich automatisch geladen werden (Menü Einstellung).

Bei „BM-Name“ können eigene Beschreibungen gespeichert werden.

Wichtig ist die Funktionsweise:

Die Konfig-Software kann beim Start aus der am PC abgelegten Datenbank den dort gespeicherten Stand sofort anzeigen. Hat sich aber an der MoBaSbS irgendwas verändert, muss über den BM-Report alles neu gescannt werden. Durch „Übernahme“ wird der Stand in den RAM der laufenden Konfig-Software gelegt, durch „speichern“ im Menü in die PC-Datenbank geschrieben.

Änderungen in der Software, die fest in der MoBaSbS integriert werden sollen, müssen ins EEPROM gespeichert werden.

Für den PMC wurde die Moduladresse von 31 auf 4 geändert.

USC: Schaltbefehle DCC aktiviert

Alle Aktionen immer ins EEPROM speichern und einen Reset auslösen.

Die Module werden über die Werkzeugleiste 1 angesprochen, in der 2. Leiste kann, hauptsächlich bei PM-Modulen, der Typ geändert werden (falls erforderlich).

Fremd-Dekoder können im PMC-Modus mit ihrer Adresse eingetragen werden. Sie liefern natürlich keine Infos wie Prozessortyp und Dekoder-Art.

Die Informationen werden aber in SRAM und EEPROM gespeichert.