

Het instellen van de AVR-fuses

De fuses van de AVR-microcontroller worden gebruikt om de controller te configureren en vormen de systeem-parameters. Niet alle fuses behoeven te worden ingesteld maar er zijn er een paar die van te voren ingesteld moeten worden of tenminste moeten worden gecontroleerd. Het is mogelijk dat de chip niet geprogrammeerd kan worden omdat de fuses verkeerd staan ingesteld.

Fuse Bits and Bytes

De AVR microcontrollers onderscheiden low, high and extended fuse bytes. De specifieke betekenis kan per AVR-controller enigszins verschillen. De details kunnen worden opgezocht in de betreffende databladeren. We richten ons hier op de AT90CAN32.

Enigszins verwarrend in de databladeren is dat een logische 1 betekent dat de fuse niet geprogrammeerd is (unprogrammed) en een logische nul betekent het bit geprogrammeerd is.

Voorbeeld: Maak bit 7 logisch 0 om de bit actief te maken.

Informatie over de fuse bits is te vinden in de handleiding onder 'Memory programming'. Blz. 336 in de Atmel AT90CAN32/64/128 handleiding.

Het programmeren van de Fuse Bits

Het proces van programmeren omvat:

1. De datasheet lezen om de juiste configuratie te weten te komen;
2. Het bepalen van de juiste fuse byte(s) waarde;
3. Het eigenlijke programmeren van de fuses in de controllerchip.

De Timloto AT90CAN32 controller is uitgerust met een extern crystal van 8 MHz. Omdat er ook andere mogelijkheden zijn (je kunt bijv. gebruik maken van de interne klok) moet de Clock Source (éénmalig) worden ingesteld. Hiervoor zijn 4 CKSEL fuse bits aanwezig. We lezen dat voor een extern 8 MHz crystal de CKSEL-bits 3 tot 0 op 1111 gezet moeten worden (blz 38). De CKSEL fuse bits bevinden zich in de Fuse Low byte. De nu volgende tabellen over de Fuse bytes zijn uit de AT90CAN32 handleiding overgenomen

Fuse Low Byte			
Bit Number	Name	Description	Default
7	CKDIV8	Divide clock by 8	0 (programmed)
6	CKOUT	Clock output	1 (unprogrammed, no BOD)
5	SUT1	Select start-up time	1 (unprogrammed)
4	SUT0	Select start-up time	0 (programmed)
3	CKSEL3	Select Clock source	0 (programmed)
2	CKSEL2	Select Clock source	0 (programmed)
1	CKSEL1	Select Clock source	1 (unprogrammed)
0	CKSEL0	Select Clock source	0 (programmed)

Fuse High Byte			
Bit Number	Name	Description	Default
7	OCDEN	Enable OCD	1 (unprogrammed, OCD disabled)
6	JTAGEN	Enable JTAG	0 (programmed, JTAG enabled)
5	SPIEN	Enable Serial Program and Data Downloading	0 (programmed, SPI programming enabled)
4	WDTON	Watchdog Timer always on	1 (unprogrammed)
3	EESAVE	EEPROM memory is preserved through the Chip Erase	1 (unprogrammed, EEPROM not preserved)
2	BOOTSZ1	Select Boot Size	0 (programmed)
1	BOOTSZ0	Select Boot Size	0 (programmed)
0	BOTRST	Select Reset Vector	1 (unprogrammed)

Extended Fuse Byte			
Bit Number	Name	Description	Default
7	-		1
6	-		1
5	-		1
4	-		1
3	BODLEVEL2	Brown-out Detector trigger level	1 (unprogrammed)
2	BODLEVEL1	Brown-out Detector trigger level	1 (unprogrammed)
1	BODLEVEL0	Brown-out Detector trigger level	1 (unprogrammed)
0	TAOSEL	Reserved for factory tests	1 (unprogrammed)

Onze controller moet worden ingesteld op:

extended FFh (correct)

high fuse 99h (correct)

low fuse CFh (in te stellen) (11001111)

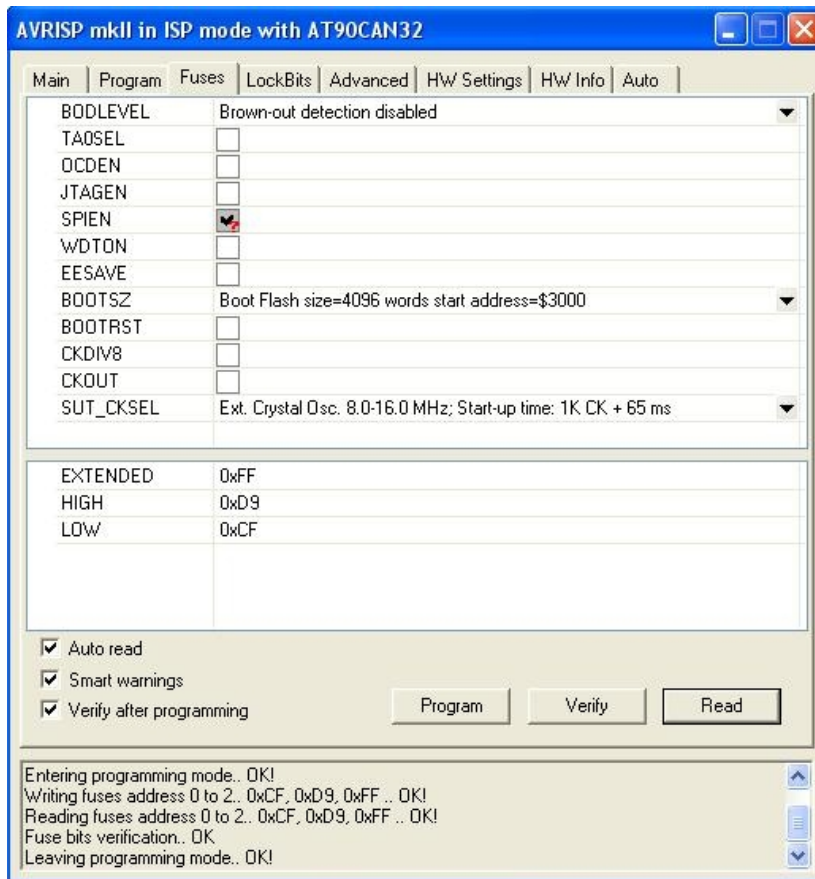
Ook wanneer we maar een paar bits zouden willen veranderen moeten we wel de hele (low) fuse byte programmeren.

Het programmeren van de Fusebyte(s)

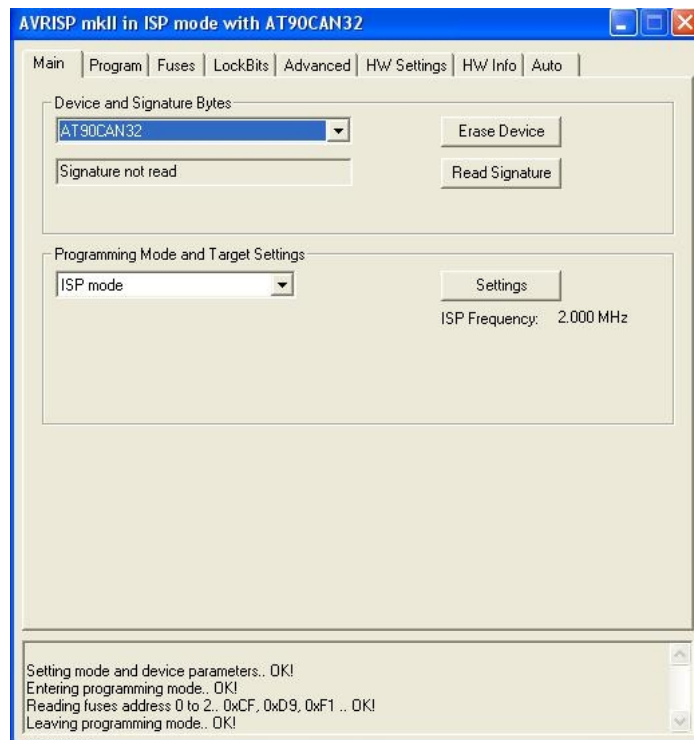
Om de fuse bytes te veranderen kunnen we de waarden uitlezen en programmeren.

Onze hardware programmer (AVRSTUDIO, AVRDUDE) is hier toe in staat.

In AVRSTUDIO gaan we naar 'Tools' en 'Program AVR'. We komen dan het nu volgende scherm tegen. We stellen alle fuses in gelijk de schermafdruk en klikken op 'program', de fuses worden dan geprogrammeerd en uitgelezen.



Denk erom dat ook JTAGEN wordt uitgeschakeld anders werkt poortF niet goed. Wanneer we ook CKDIV8 uitschakelen dan kan de programmeersnelheid op 2 Mhz worden gezet.



Wanneer we onder Linux werken en avrdude gebruiken dan kunnen we de fuses met de volgende commandoregel in de files 'hfuse' en 'lfuse' uitlezen (denk om sudo su) :

```
...# avrdude -p c32 -c avrispmkII -B0.5 -P usb: xx -U hfuse:r:hfuse:h  
-U lfuse:r:lfuse:h
```

```
erasing chip  
writing output file 'hfuse'  
writing output file 'lfuse'
```

```
avrdude done. Thank you.
```

We kunnen de nieuwe waarde van de lfuse naar de controller sturen met de volgende regel

```
...# avrdude -p c32 -c avrispmkII -B0.5 -P usb: xx -U lfuse:w:0xCF:m
```

Informatie

De fuse bytes worden niet gewist wanneer we het flash-geheugen opnieuw programmeren zodat het niet nodig is om elke keer de fuses opnieuw in te stellen. Dit houdt natuurlijk ook in dat wanneer ze verkeerd zouden staan de chip niet meer seriëel programmeerbaar is.

Het is bijv. mogelijk om de SPIEN fuse in te stellen. Als je dit doet dat kun je de chip niet meer via de ISP bereiken (herstel dient dan te geschieden door parallel te programmeren (STK500).

Niet alle fuses kunnen door ISP-programmering worden veranderd. Parallel programmering is dan vereist. De lock bits fuses kunnen de chip blokkeren. Niet aankomen dus.

Je kunt ook de -v optie in avrdude gebruiken wanneer je de fuse bytes wilt uitlezen. De 'verbose' geeft veel nuttige extra informatie. Hier is de uitlezing van de AT90CAN32:

```
..# avrdude -v -p c32 -c avrispmkII -B0.5 -P usb:xx -e -U hfuse:r:high.txt:r -U  
lfuse:r:low.txt:r
```

```
avrdude: Version 5.5, compiled on Nov 10 2007 at 00:07:28  
Copyright (c) 2000-2005 Brian Dean, http://www.bdmicro.com/
```

```
System wide configuration file is "/etc/avrdude.conf"  
User configuration file is "/root/.avrduderc"  
User configuration file does not exist or is not a regular file, skipping
```

```
Using Port          : usb:  
Using Programmer   : avrispmkII  
Setting bit clk period: 0.5
```

```
avrdude: usbdev_open(): Found AVRISP mk2 (CC2), serno: 0000A00128255
```

```
AVR Part           : AT90CAN32  
Chip Erase delay   : 9000 us  
PAGEL              : PD7  
BS2                : PA0  
RESET disposition  : dedicated  
RETRY pulse        : SCK  
serial program mode : yes  
parallel program mode : yes  
Timeout            : 200  
StabDelay          : 100  
CmdexeDelay        : 25  
SyncLoops          : 32  
ByteDelay          : 0  
PollIndex          : 3  
PollValue          : 0x53  
Memory Detail      :
```

Memory	Type	Mode	Block Poll			Paged	Page			Polled		
			Delay	Size	Indx		Size	Size	#Pages	MinW	MaxW	ReadBack
EEPROM		65	20	8	0	no	1024	8	0	9000	9000	0xff 0xff
Flash		65	6	256	0	yes	32768	256	128	4500	4500	0xff 0xff
LFUSE		0	0	0	0	no	1	0	0	9000	9000	0x00 0x00
HFUSE		0	0	0	0	no	1	0	0	9000	9000	0x00 0x00
EFUSE		0	0	0	0	no	1	0	0	9000	9000	0x00 0x00
Lock		0	0	0	0	no	1	0	0	9000	9000	0x00 0x00
Calibration		0	0	0	0	no	1	0	0	0	0	0x00 0x00
Signature		0	0	0	0	no	3	0	0	0	0	0x00 0x00

```

Programmer Type : STK500V2
Description      : Atmel AVR ISP mkII
Programmer Model: AVRISP mkII
Hardware Version: 0
Firmware Version: 9.09
Vtarget         : 5.0 V
SCK period      : 0.50 us

```

avrdude: AVR device initialized and ready to accept instructions

Reading | ##### | 100% 0.01s

```

avrdude: Device signature = 0x1e9581
avrdude: safemode: lfuse reads as CF
avrdude: safemode: hfuse reads as 99
avrdude: safemode: efuse reads as FF
avrdude: erasing chip
avrdude: reading hfuse memory:

```

Reading | ##### | 100% 0.00s

```

avrdude: writing output file "hfuse"
avrdude: reading lfuse memory:

```

Reading | ##### | 100% 0.00s

```

avrdude: writing output file "lfuse"

```

```

avrdude: safemode: lfuse reads as CF
avrdude: safemode: hfuse reads as 99
avrdude: safemode: efuse reads as FF
avrdude: safemode: Fuses OK

```

avrdude done. Thank you.