ULTIMATE Arduino Uno Hardware Manual

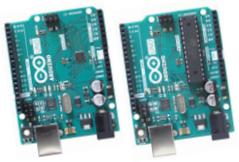
Een hoofdstuk als voorbeeld: Main Microcontroller Bootloader

Warwick Smith (Zuid Afrika)

In deze aflevering van Elektor Books presenteren we een deel van een hoofdstuk uit het boek *Ultimate Arduino Uno Hardware Manual* van Warwick A. Smith. Dit boek is door Elektor gepubliceerd in juni 2021. Je komt de Arduino Uno tegenwoordig bijna overal tegen. Hij is ook goedkoop en daarom is het gemakkelijk om belangrijke eigenschappen van de hardware over het hoofd te zien. En dat kan tot problemen leiden bij het ontwikkelen van eigen toepassingen. Problemen met de Uno Bootloader en de Fuses brengen nog steeds veel lezers in verwarring en scoren hoog in de top-10 van veel gestelde vragen bij Elektor Lab. Daarom hierbij enkele “ultieme” antwoorden uit dit nieuwe Elektor-boek.

De bootloader

Als de microcontroller (de ATmega328P) op een Arduino Uno ooit wordt vervangen door een exemplaar waar nog geen bootloader in zit, zullen we er zelf een bootloader op moeten zetten. Dat is te doen met behulp van Microchip Studio en een externe USB-programmer die wordt aangesloten op de ICSP-header. Het programmeren gaat op dezelfde manier als bij de ATmega16U2, maar we moeten de ICSP-header gebruiken die aan het andere eind van de kaart zit als de USB-connector. We kunnen langs die weg ook een backup maken van de bootloader die al op een ATmega328P zit. In dit artikel bespreken we hoe dat in zijn werk gaat, en gaan we ook in op de fuse-instellingen van de ATmega328P op een Arduino Uno.



De bootloader kan ook worden hersteld vanuit de Arduino-IDE, maar niet als Microchip Studio is geïnstalleerd, want dan krijgen we een conflict tussen de drivers.

Een backup maken van de firmware in de ATmega328P met Microchip Studio

We kunnen als voorzorgsmaatregel de firmware op een ATmega328P op de Arduino Uno bewaren in HEX- en BIN-files met de programmeer-utility in Microchip Studio. Dat is zeker van belang, als we werken met een gekloonde kaart, die misschien wel een andere bootloader heeft dan wordt gebruikt op een authentieke Arduino Uno.
Sluit een geschikte USB-programmer aan op de computer en verbind de kabel met de zespolige stekker met de ICSP-header op de Uno. In de penconfiguratie van de ICSP-header voor de ATmega328P is te vinden waar pen 1 zit. Voed de Arduino Uno via USB of via een externe voeding.
Start Microchip Studio en open de programmeer-utility door te klikken op het pictogram *Device Programming* in de werkbalk, of kies *Tools Device Programming* uit het menu van Microchip Studio. Kies in het dialoogvenster *Device Programming* de juiste USB-programmer onder *Tools*, kies de *ATmega328P* als *Device*, controleer of *ISP* is geselecteerd onder *Interface* en klik dan op de button *Apply*. Klik op de knop *Read* onder *Device Signature* om te controleren of alles goed is ingericht en dat Microchip Studio met het target-apparaat kan communiceren. Als alles in orde is, worden de identificatie en de spanning van de target dan weergegeven in het dialoogvenster *Device Programming*.
Klik in de linker kolom van dit dialoogvenster op *Memories*. Klik dan op de knop *Read...* bij *Flash* in het dialoogvenster *Memories* en ga naar een map waar u de HEX-file uit het flashgeheugen van de ATmega328P wilt gaan opslaan. Voer onderin het opslagdialoogvenster de naam in die u de HEX-file wilt geven. Die naam kunt u zelf kiezen. Klik daarna op de knop *Save*. De *Device Programming*–utility leest dan de inhoud van het flashgeheugen van de ATmega328P uit en slaat die op in een HEX-bestand met de gekozen naam. Met dit HEX-bestand kan de originele firmware van de ATmega328P worden hersteld als dat nodig is. Het HEX-bestand bevat de hele inhoud van het flashgeheugen, inclusief de sketch die de gebruiker in het flashgeheugen had staan op het moment van het maken van het HEX-bestand.

Optiboot bootloader-firmware

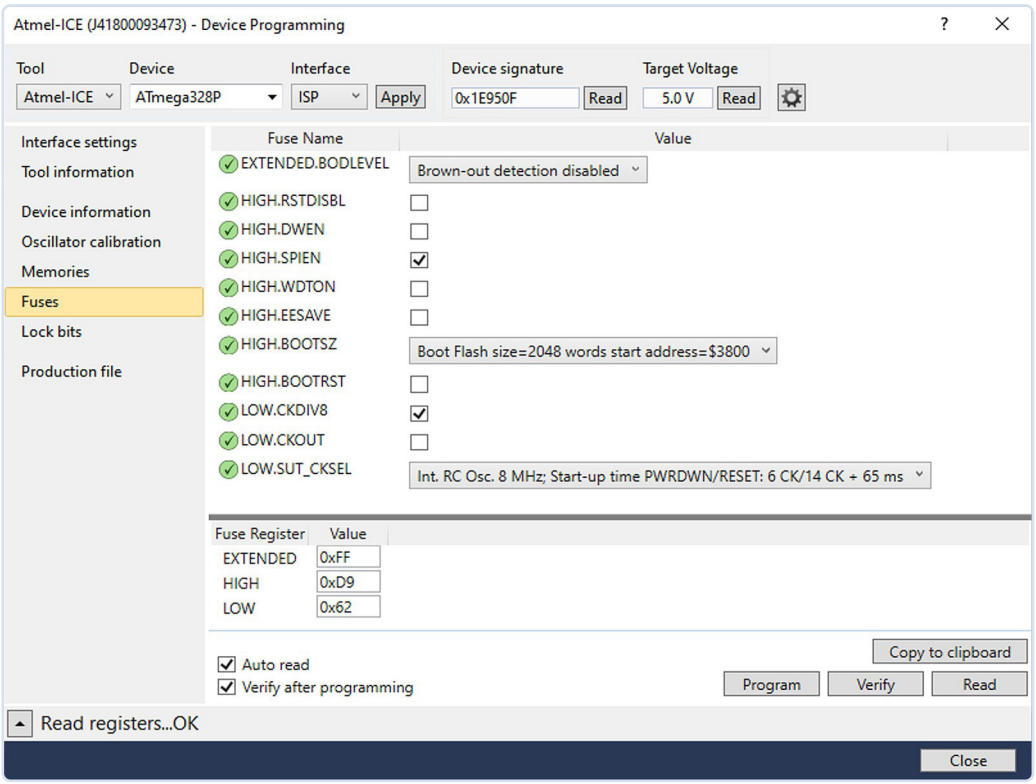
De Arduino Uno gebruikt tegenwoordig de **Optiboot**-bootloader in de ATmega328P-microcontroller, die in de plaats is gekomen van een oudere bootloader. Optiboot gebruikt een geheugensegment van 512 bytes in het flashgeheugen, terwijl de oudere bootloader 2 k gebruikte. Dat betekent dat er nu 1,5 k meer flashgeheugen beschikbaar is voor de gebruiker.
De broncode in C en de HEX-files van Optiboot zijn bij de Arduino-IDE te vinden in de map:
`arduino-1.8.13\hardware\arduino\avr\bootloaders\optiboot`
waarbij de naam van de basismap overeenkomt met de versie van de Arduino-IDE. Dezelfde files zijn ook te vinden op de GitHub pagina's van Arduino [1].
Het Optiboot-project staat op GitHub bij *github.com/optiboot/optiboot* en de Optiboot-wiki is te vinden op *github.com/optiboot/optiboot/wiki*.
Optiboot laat de L-LED van de Arduino opflitsen als het start. Daarna start het de sketch van de gebruiker, of het wacht op een nieuwe sketch als de UNO is gereset door de Arduino-IDE. Zie de Optiboot-Wiki [2] voor meer details.

De bootloader herstellen

We kunnen de Optiboot-bootloader van een ATmega328P herstellen, of we kunnen Optiboot programmeren op een maagdelijke ATmega328P-microcontroller. Ook dat gaat weer met Microchip Studio en een externe USB-programmer. Verbind de ATmega328P met Microchip Studio op dezelfde manier als onder "Een backup maken van de firmware in de ATmega328P met Microchip Studio" Sluit de USB-programmer aan op de ICSP-header zoals eerder beschreven. Open de *Device Programming*-utility in Microchip Studio en maak verbinding met de programmer en de ATmega328P. Klik in de linker kolom van de *Device Programming*-dialoog op *Memories*. Klik op de button *[...]* bij *Flash*. Er verschijnt dan een dialoogvenster waarin we naar de HEX-file kunnen navigeren die we op de ATmega328P willen laden. We kunnen het HEX-bestand gebruiken dat we eerder hebben gemaakt toen we een backup van de bootloader maakten, of we kunnen een verse bootloader van het Internet halen op:
`arduino-1.8.13\hardware\arduino\avr\bootloaders\optiboot`
Voor de Arduino Uno is dat *optiboot_atmega328.hex*. Selecteer het juiste HEX-bestand en klik dan op *Open*. U komt dan terug in de dialoog *Device Programming*. Klik nu op de knop *Program* bij *Flash*. Als alles goed is ingesteld, wordt het flashgeheugen dan geprogrammeerd met het geselecteerde HEX-bestand.
Als de bootloader op een maagdelijk ATmega328P is geladen, moeten we ook de fuses nog instellen, we komen daar zometeen op terug. Als de fuses al goed staan, kunnen we nu de dialoog *Device Programming* in Microchip Studio sluiten en de voeding van de Arduino Uno losnemen. Ontkoppel de USB-programmer van de Arduino en schakel de Arduino daarna weer in. Als de fuses al goed staan, kunt u de nieuw geladen bootloader testen door vanuit de Arduino-IDE een test-sketch naar de Arduino te sturen via de USB-poort.

Fuse-instellingen van de ATmega328P

Op een nieuwe, lege ATmega328P moeten we eerst de Optiboot-bootloader programmeren, zoals hierboven is beschreven. Daarna moeten we de ATmega328P-fuses instellen voor gebruik in de Arduino Uno. We kunnen de fuses uitlezen en instellen met behulp van de Device Programming-utility in Microchip Studio. Om dat te doen, sluiten we eerst weer een externe USB-programmer aan op de ICSP-header, zoals eerder beschreven. Open in Microchip Studio de *Device Programming*-utility, kies de juiste instellingen voor uw tool, device en interface, zoals al eerder is beschreven en klik dan op *Apply* om verbinding te maken. Klik in de linker kolom van het dialoogvenster *Device Programming* op *Fuses*.
In **figuur 1** ziet u, wat u kunt verwachten als u kijkt naar de fuse-instellingen van een nieuwe, lege ATmega328P-microcontroller. De standaardwaarden van de fuse-registers zijn als volgt.



Figuur 1: Standaard fuse-instellingen voor een nieuwe, lege ATmega328P.

EXTENDED	0xFF
HIGH	0xD9
LOW	0x62

Deze fuses moeten worden geprogrammeerd op de waarden die u ziet in **figuur 2** en **tabel 1**. De volgende fuses mogen niet op de standaardwaarden blijven staan:

Fuse	**Verander in:**
EXTENDED.BODLEVEL	Brownout-detectie bij VCC=2,7V
HIGH.EESAVE	(optioneel, zie volgende pagina voor uitleg)
HIGH.BOOTSZ	Boot Flash-grootte =256 woorden; startadres=$3F00
HIGH.BOOTRST	aangevinkt
LOW.CKDIV8	niet aangevinkt
LOW.SUT_CKSEL	Ext. Kristalosc. 8,0 MHz; opstarttijd PWRDWN/RESET: 16K/14 CK + 65ms

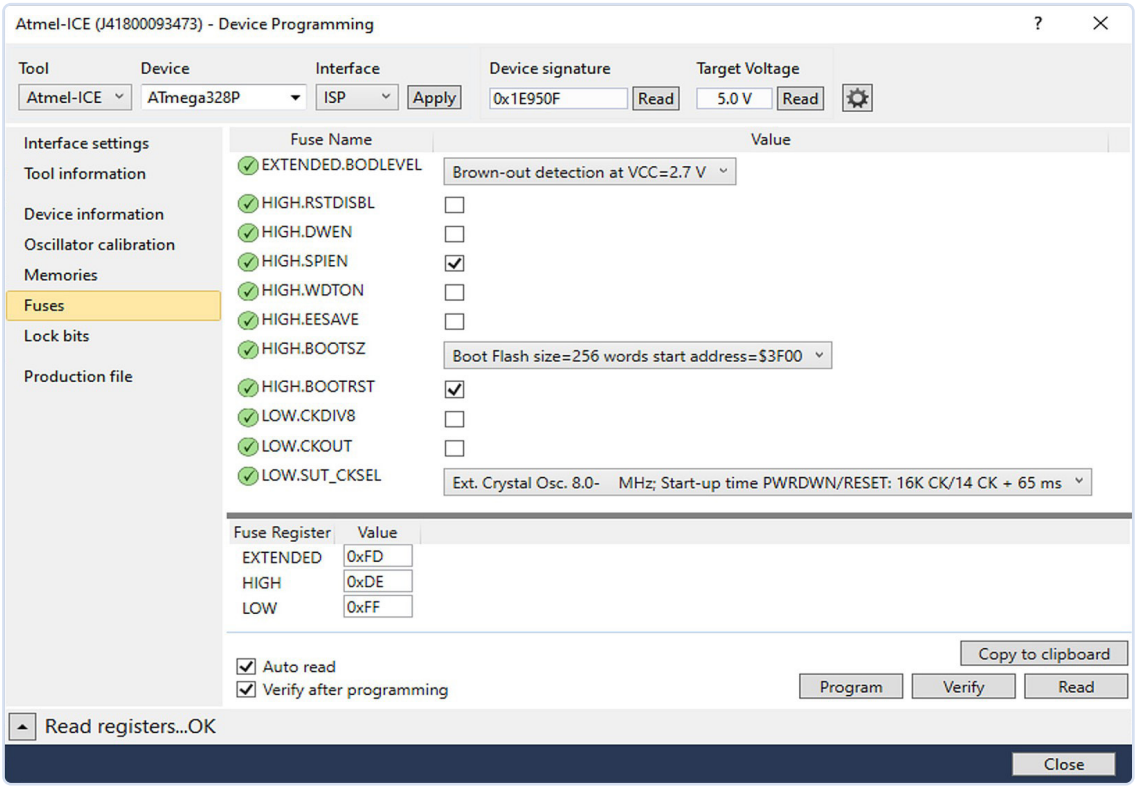
Als deze instellingen zijn gedaan, zullen de fuse-registerwaarden onderaan als volgt zijn:

EXTENDED	**0xFD**
HIGH	0xDE (EESAVE niet aangevinkt) of 0xD6 (EESAVE aangevinkt)
LOW	0xFF

Het is gebleken, dat EESAVE bij oudere Arduino Uno R3-kaarten was aangevinkt, maar niet bij nieuwere exemplaren. Als EESAVE (HIGH.EESAVE-fuse) is aangevinkt in de dialoog, beschermt dat de inhoud van het EEPROM-geheugen als de chip wordt gewist.
Als u de aanpassingen hebt gedaan, klik dan op de knop *Program* onderin het dialoogvenster. Dat programmeert de nieuwe fuse-instellingen in de ATmega328P. Klik *OK* in het waarschuwingsvenster om door te gaan. Sluit het *Device Programming*-venster, neem de voeding van de Arduino Uno los en koppel de USB-programmer af. Sluit de Arduino Uno aan op de computer via de USB-poort en laad een sketch vanuit de Arduino-IDE om te controleren dat de bootloader werkt en de fuse-instellingen goed zijn.

De jumper RESET-EN

Sommige AVR USB-programmers hebben debug-mogelijkheden. Dat zijn de AVR Dragon (die niet meer wordt verkocht door Microchip) en de Atmel-ICE (die oudere USB-programmers vervangt). De AVRISP mkII (die niet meer wordt verkocht door Microchip) heeft geen debug-mogelijkheden; het is alleen een programmer. Dat geldt ook voor de vele hobby-programmers die in omloop zijn, zoals de USBasp en de USBtinyISP, ook dat zijn alleen programmers. Met debug-mogelijkheden wordt bedoeld: de mogelijkheid van de USB-programmer/debugger om in combinatie met software zoals Microchip Studio naar de inhoud van het geheugen en de interne registers van de microcontroller op de Arduino te kijken, de code stap-voor-stap uit te voeren en breakpoints te zetten. Die mogelijkheid is niet beschikbaar in de Arduino-IDE versie 1.8.13, wat op dit moment de gangbare versie is.
Om de debug-mogelijkheden van de USB-programmer/debugger te gebruiken, wordt die verbonden met de ICSP-header, net als bij het programmeren van het flashgeheugen of bij het lezen van de fuse-instellingen. We kunnen dan een C of C++-programma in de



Figuur 2: Fuse-instellingen voor de ATmega328P op een Arduino Uno.

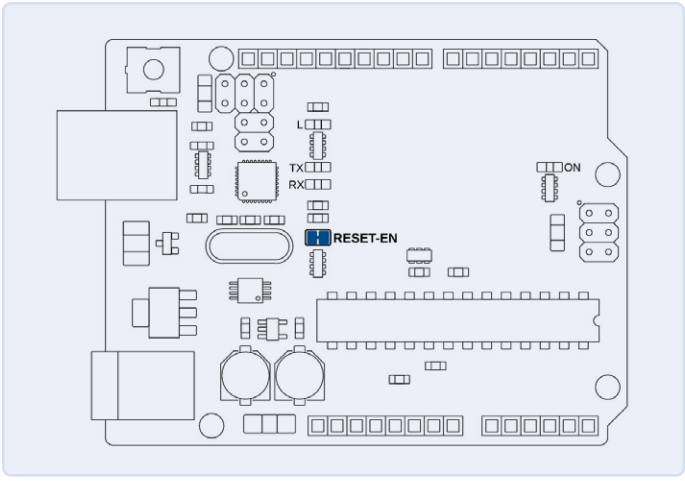
Tabel 1: Fuse-instellingen voor de ATmega328P op de Arduino Uno.

Naam	Waarde
EXTENDED.BODLEVEL	Brownout-detectieniveau op VCC=2,7 V
HIGH.RSTDISBL	niet aangevinkt
HIGH.DWEN	niet aangevinkt
HIGH.SPIEN	☑
HIGH.WDTON	niet aangevinkt
HIGH.EESAVE	niet aangevinkt op moderne boards
HIGH.BOOTSZ	Boot Flash-grootte =256 woorden; startadres=$3F00
HIGH.BOOTRST	☑
LOW.CKDIV8	niet aangevinkt
LOW.CKOUT	niet aangevinkt
LOW.SUT_CKSEL	Ext.kristaloscillator 8,0 MHz; Start-up PWRDWN/RESET 16K CK/14 CK + 65 ms

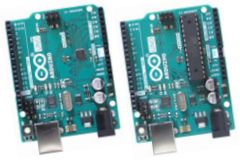
AVR laden met Microchip Studio. En daarna kunnen we de code debuggen vanuit Microchip Studio.

Het debugsysteem op de ATmega328P-microcontroller staat bekend als **debugWIRE**. We kunnen daarmee de debug-software koppelen met de target-microcontroller via de ICSP-header met een USB-programmer die debugging ondersteunt. Om debugWIRE te laten werken op de ATmega328P op Arduino Uno-boards, moet het reset-circuit worden afgekoppeld van de resetpen. Arduino Uno's hebben een soldeerlink met het label RESET-EN, die bestaat uit twee soldeereilanden, verbonden door een printspoor. Het spoor tussen die soldeereilanden moet worden doorgesneden om een deel van het resetcircuit af te koppelen van de resetpen als debugWIRE wordt gebruikt. De verbinding kan worden hersteld door de twee soldeereilanden aan elkaar te solderen.

In **figuur 3** ziet u de locatie van de RESET-EN-soldeereilanden op de Arduino Uno. Deze functie wordt alleen gebruikt door geavanceerde Arduino-gebruikers, die de kaart programmeren met behulp van pure C-code en Microchip Studio. Lezers die geïnteresseerd zijn in het leren van C om Arduino's te programmeren, kunnen terecht bij het boek *C Programming with Arduino* (ISBN 978-1-907920-46-2) van dezelfde auteur, dat is gepubliceerd door Elektor International Media. Dat boek heeft een hoofdstuk over het debuggen van



Figuur 3: Locatie van de RESET-EN -jumper op een Arduino Uno.



Arduino Uno- en Mega 2560-boards met behulp van Atmel Studio, dat inmiddels is vervangen door Microchip Studio.

Alternatieve manieren om de firmware te programmeren

Alternatieve manieren om de bootloader in een Arduino Uno te programmeren, zijn te vinden op de Arduino playground in de Bootloader-sectie [3]. Daar staat ook informatie over het gebruik van een tweede Arduino als programmer en over alternatieve programmers.
Het gebruik van een Arduino als een ISP (In-System Programmer) wordt ook beschreven op de Arduino-website [4]. In hoofdstuk 2, sectie 2.8 van het boek is meer informatie te vinden over het laden van sketches op de Arduino Uno met behulp van een externe programmer en de Arduino-IDE. U vindt daar ook informatie over het herstellen van de bootloader in de microcontroller met behulp van een externe programmer en de Arduino-IDE.
Bij [5] vindt u informatie over het DFU-programmeren de ATmega16U2.

210345-03

Opmerking van de redactie: Dit artikel is een deel van een hoofdstuk uit het boek Ultimate Arduino Uno Hardware Manual, dat we enigszins hebben bewerkt om het in het formaat van Elektor Magazine te passen. Omdat dat een gedeelte is uit een grotere publicatie, kunnen er in dit artikel verwijzingen voorkomen naar andere hoofdstukken in het boek. De auteur en de redactie hebben hun best gedaan om dit te vermijden en zijn altijd bereid om op vragen in te gaan. U vindt hun contactgegevens in het tekstkader **Vragen of opmerkingen**?

GERELATEERDE PRODUCTEN

- **W. A. Smith, *Ultimate Arduino Uno Hardware Manual* (Elektor 2021) (gedrukt) (SKU 19678)**
  www.elektor.nl/19678
- **W. A. Smith, *Ultimate Arduino Uno Hardware Manual* (Elektor 2021) (e-boek) (SKU 19679)**
  www.elektor.nl/19679
- **W. A. Smith, *C Programming with Arduino* (Elektor 2018) (gedrukt) (SKU 17574)**
  www.elektor.nl/17574

- **W. A. Smith, *C Programming with Arduino* (Elektor 2018) (e-boek) (SKU 18209)**
  www.elektor.nl/18209

Vragen of opmerkingen?
Als u technische vragen of opmerkingen hebt over dit artikel, Stuur dan email naar de auteur via warwsmi@axxess.co.za of naar de redactie via editor@elektormagazine.com.

Met bijdragen van:
Tekst en afbeeldingen: **Warwick A. Smith**
Redactie: **Jan Buiting**
Lay-out: **Sylvia Sopamena**
Vertaling: **Evelien Snel**

WEBLINKS

[1] Arduino Core AVR Tree: http://www.github.com/arduino/ArduinoCore-avr/tree/master/bootloaders/optiboot
[2] Optiboot-Wiki: http://www.github.com/Optiboot/optiboot/wiki/HowOptibootWorks
[3] Arduino Bootloader: https://playground.Arduino.cc/Main/ArduinoCoreHardware/#Bootloader
[4] Arduino als een ISP: https://www.arduino.cc/en/Tutorial/BuiltInExamples/ArduinoISP
[5] DFU-programmeren van de ATmega16U2: https://www.arduino.cc/en/Hacking/DFUProgramming8U2