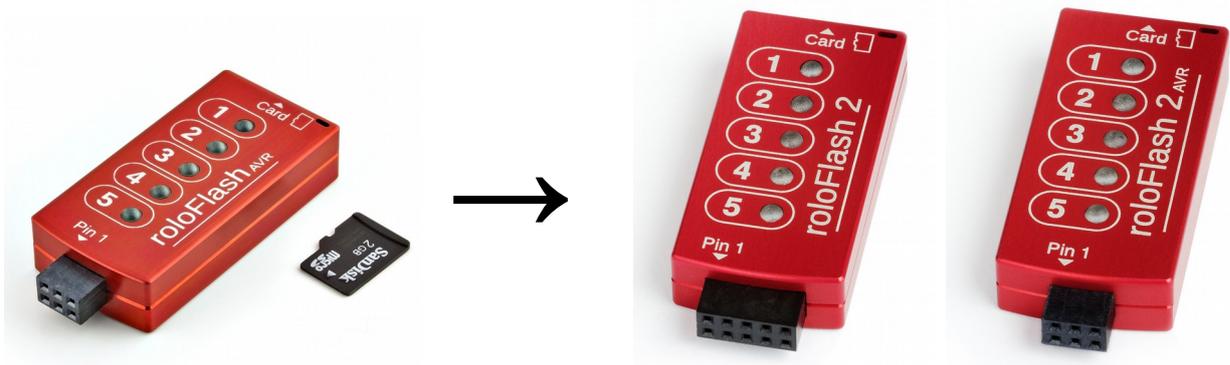


halec
Herrnröther Str. 54
63303 Dreieich
Germany

www.halec.de



Migrations-Leitfaden roloBasic-Skripte für roloFlash AVR und roloFlash-2-Familie



Anleitung zur Migration von roloBasic-Skripten von roloFlash AVR auf die
roloFlash-2-Familie (roloFlash 2 und roloFlash 2 AVR)

Dokumentenversion 1.0.1 vom 2019-03-12
(Stand der Software: 03.AA)

Copyright © 2009-2019 halec. Alle Marken, Logos und Bilder sind Eigentum der jeweiligen Hersteller bzw. Urheber. Änderungen und Irrtümer vorbehalten.

Inhaltsverzeichnis

I	RoloFlash: API-Versionen.....	iii
II	Konzeptionelle Unterschiede.....	iv
III	Einzelne Funktionen.....	vi

I roloFlash: API-Versionen

Folgende roloFlash-API-Versionen sind zur Zeit im Umlauf:

- Für roloFlash AVR:
 - (nur eine Version im Umlauf)

- Für die roloFlash-2-Familie:
 - 02
 - 03

Sie erkennen die Version an der Major-Version der Software der roloFlash-2-Familie:

Bsp. „03.AA“: API-Version 3 (Major-Version = „03“, Minor-Version = „AA“)

Diese Anleitung zeigt Ihnen, wie Sie Ihre roloBasic-Skripte von roloFlash AVR auf die roloFlash-2-Familie in der API-Version 3 anpassen.

Hinweis:

Auf der zum roloFlash gelieferten microSD-Karte befinden sich im Verzeichnis „scripts“ zahlreiche Beispiel-Skripte für verschiedene Abläufe und verschiedene Mikrocontroller, die evtl. ein besserer Anlaufpunkt zur Konvertierung Ihrer Skripte sein könnten als diesem Migrations-Leitfaden zu folgen.

Hinweis:

Wenn Sie Hilfe benötigen bei der Anpassung Ihrer roloBasic-Skripte

- von roloFlash AVR auf die roloFlash-2-Familie oder
- von einer roloFlash-2-API-Version auf eine andere,

kontaktieren Sie uns bitte per E-Mail an [<rh@halec.de>](mailto:rh@halec.de).

II Konzeptionelle Unterschiede

roloFlash AVR	roloFlash-2-Familie
<p>Ein roloBasic-Skript kann ein Magic-Cookie in der ersten Zeile haben. Dieses muss dann als Kommentar kodiert sein:</p> <pre>!roloFlash AVR</pre>	<p>Ein roloBasic-Skript muss ein Magic-Cookie in der ersten Zeile haben, Dieses muss mit einem „#“ beginnen und lautet zum Beispiel:</p> <pre>#roloFlash 2, v03.AA</pre> <ul style="list-style-type: none"> • Bitte schreiben Sie auch bei roloFlash 2 AVR „#roloFlash 2“, nicht „roloFlash 2 AVR“ • Dabei ist v03 die API-Version, die der Major-Version der roloFlash-Software entspricht • Dabei ist das „AA“ die Minor-Version der roloFlash-Software und dient nur zur Dokumentation. Sie können hier auch ein „*“ als Wildcard angeben: <pre>#roloFlash 2, v03.*</pre>
<p>Unterstützt nur eine Familie an Targets, die immer über ISP angeschlossen sind.</p> <p>Auswirkung auf das API: - Sofortige Kommunikation mit dem Target möglich (z.B. getSignature)</p>	<p>Unterstützt mehrere Familien an Targets, die über unterschiedliche Busse angeschlossen sind. Bei manchen Bussen können auch mehrere Targets angeschlossen sein (JTAG für STM32).</p> <p>Auswirkung auf das API: - Zuerst muss ein Bus geöffnet werden (z.B. busHandle = bus_open(ISP, 100000), man erhält ein Bus-Handle. - Dann muss das Target unter Angabe des Bus-Handles geöffnet werden, z.B. mit targetHandle = target_open(busHandle, 0) - Erst jetzt kann mit dem Target unter Angabe des Target-Handles kommuniziert werden, z.B. target_getDeviceId(targetHandle)</p>
<p>getSignature bestimmt implizit auch Parameter wie Flashsize und FlashPagesize</p>	<p>getDeviceID hat keine Seiteneffekte.</p> <p>Die nötigen Parameter können aus einer internen Datenbank ausgelesen werden und müssen für das Flashen explizit gesetzt werden:</p>

```
dbHandle = DB_getHandle(<targetName>)
Beispiel:
    dbHandle = DB_getHandle(Atmega128)

flashSize = DB_get(dbHandle,
DB_FLASHSIZE)

pageSize = DB_get(dbHandle,
DB_FLASHPAGESIZE)

target_setMemoryMap targetHandle, FLASH,
MEM_SIZE, flashSize

target_setMemoryMap targetHandle, FLASH,
MEM_PAGESIZE, pageSize
```

III Einzelne Funktionen

roloFlash AVR	roloFlash-2-Famile
targetPresent	target_getPresent
programTarget	target_setMode targetHandle, PROGRAMMODE
runTarget	target_setMode targetHandle, RUNMODE
restartTarget	target_restart
SetProgrammingSpeed <speed>	bus_open(ISP, <speed> oder bus_setSpeed <speed>
getTargetVoltage	getTargetBoardVoltage
readBits	target_readBits
writeBits	target_writeBits
getSignature	target_getDeviceID
getFlashLayout	target_getMemoryMap(<targetHandle>, FLASH, MEM_SIZE) und target_getMemoryMap(<targetHandle>, FLASH, MEM_PAGESIZE)
setFlashLayout(<size>, <pagesize>)	target_setMemoryMap <targetHandle>, FLASH, MEM_SIZE, <size>) und target_setMemoryMap <targetHandle>, FLASH, MEM_PAGESIZE, <pagesize>)
getEepromLayout	target_getMemoryMap(<targetHandle>, EEPROM, MEM_SIZE) und target_getMemoryMap(<targetHandle>, EEPROM, MEM_PAGESIZE)
setEepromLayout(<size>, <pagesize>)	target_setMemoryMap <targetHandle>, EEPROM, MEM_SIZE, <size>) und target_setMemoryMap <targetHandle>, EEPROM, MEM_PAGESIZE, <pagesize>)

setExtendedAddressMode	target_setExtendedAddressMode
clearMemoryLayout	target_clearMemoryLayout
eraseFlash	target_eraseFlash
writeFileToFlash 0, <filename>	target_writeFromFile <targetHandle>, 0, <filename>, HEX, FLASH, WRITEONLY
writeVerifyFileToFlash 0, <filename>	target_writeFromFile <targetHandle>, 0, <filename>, HEX, FLASH, WRITEVERIFY
verifyFileToFlash 0, <filename>	target_writeFromFile <targetHandle>, 0, <filename>, HEX, FLASH, VERIFYONLY
writeFileToEeprom 0, <filename>	target_writeFromFile <targetHandle>, 0, <filename>, HEX, EEPROM, WRITEONLY
writeVerifyFileToEeprom 0, <filename>	target_writeFromFile <targetHandle>, 0, <filename>, HEX, EEPROM, WRITEVERIFY
verifyFileToEeprom 0, <filename>	target_writeFromFile <targetHandle>, 0, <filename>, HEX, EEPROM, VERIFYONLY
Alle Funktionen des Dateisystems, zum Beispiel: fsOpen ...	fs_open ...
Alle LED-Funktionen, zum Beispiel: ledOn ...	led_on ...