How to write SD card image

The SD card image can be written to a block device like eMMC of iMX8M(M, N, Q, P) based board in different ways. The following options from the bootloader are possible: USB mass storage (UMS), NXP Universal Update Utility (UUU) and F&S update function.

USB Mass Storage

In U-Boot it is possible to export available block devices like USB, SD Card or eMMC as USB mass storage devices with ums command:

```
PicoCoreMX8MM # help ums
ums - Use the UMS [USB Mass Storage]

Usage:
ums <USB_controller> [<devtype>] <dev[:part]>  e.g. ums 0 mmc 0
    devtype defaults to mmc
PicoCoreMX8MM #
```

The command runs in the loop and U-Boot command line is blocked until Ctrl-C would be entered.

On PicoCoreMX8M(M,N) the USB device has index 0. SD card index is 0 too and eMMC is 2. So the eMMC device can be exported over connected USB device with command:

```
PicoCoreMX8MM # ums 0 mmc 2
UMS: LUN 0, dev 2, hwpart 0, sector 0x0, count 0x748000
Device
```

Now on the host side a usb block device should be mounted. In this case tools like dd can be used to write sd card image to the mounted block device.

```
PC Console $> dd if=image.sdcard of=/dev/sd[x] conv=fdatasync
```

Other open source tool can be Win32 Disk Imager.

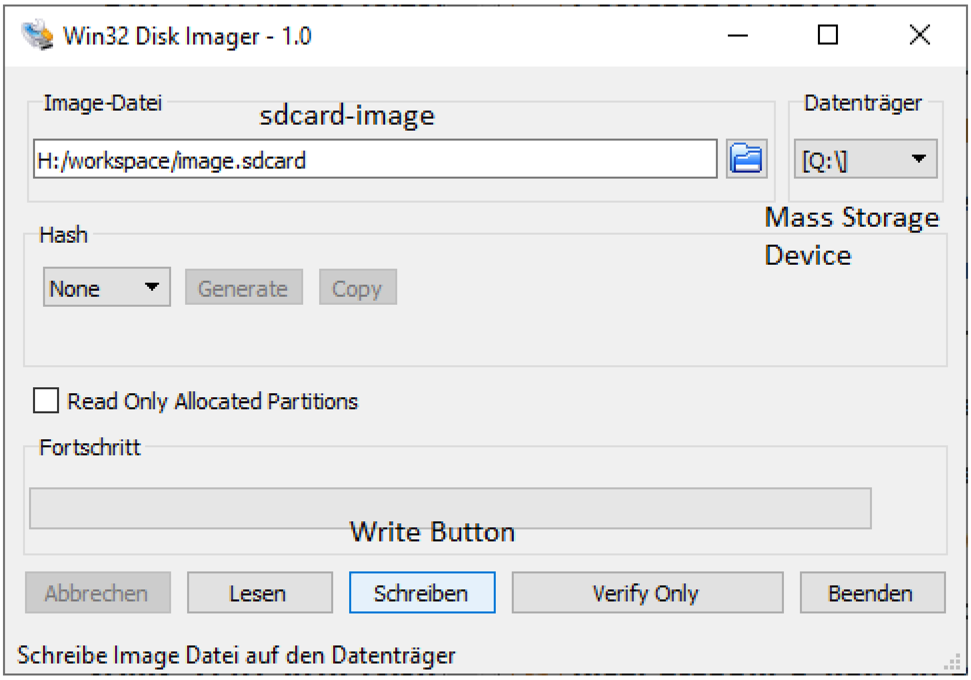
Win32 Disk Imager - 1.0

Image-Datei: H:/workspace/image.sdcard

Datenträger: [Q:\]

Hash: None

□ Read Only Allocated Partitions

Schreibe Image Datei auf den Datenträger

sdcard-image

Mass Storage Device

Write Button

Please take attention to write data to correct USB block device.

NXP Universal Update Utility

UUU is Freescale/NXP iMX Chip image deploy tools and can be downloaded from github https://github.com/NXPmicro/mfgtools.

UUU uses fastboot protocol to flash images. On the board side fastboot mode can be started with command:

```
PicoCoreMX8MM # setenv bootcmd fastboot 0; boot
Device
```

Now you can use the uuu tool on the host side to write the sd card image over connected USB device.

```
PC Console $> uuu.exe write-sdcard-image.auto
```

```
uuu_version 1.0.1

FB: ucmd setenv fastboot_dev mmc
FB: ucmd mmc dev ${mmcdev}

FB: flash -raw2sparse all image.sdcard

FB:done
```

Figure 1: Script write-sdcard-image.auto

After command execution copy process should be started:

```
PicoCoreMX8MM # setenv bootcmd fastboot 0; boot
Device
switch to partitions #0, OK
mmc2(part 0) is current device
Starting download of 16776232 bytes
................................................................................
.........................................................
downloading of 16776232 bytes finished
writing to partition 'all'
sparse flash target is mmc:2
writing to partition 'all' for sparse, buffer size 16776232
Flashing sparse image at offset 0
Flashing Sparse Image
........ wrote 16776192 bytes to 'all'
Starting download of 16776244 bytes
```

F&S Update

F&S Update functionality is described in documentation “Linux on F&S Boards” (LinuxOnFSBoards_eng.pdf) section 6.4.5. For example update script update.txt **Error! Reference source not found.** shows how 800MB data can be copied from USB stick to eMMC memory.

```
# Set MMC device
mmc dev 2
# sdcard image is ca. 800MB
# clone first chunk (400MB)
# 1. Erase first 400MB
mmc erase 0 0xc8000
# 2. read 400MB from usb
usb read ${loadaddr} 0 0xc8000
# 3. Write first chunk
mmc write ${loadaddr} 0x0 0xc8000
# 4. Erase second 400MB
mmc erase 0xc8000 0xc8000
# 5. read 400MB from usb
usb read ${loadaddr} 0xc8000 0xc8000
# 6. Write second chunk
mmc write ${loadaddr} 0xc8000 0xc8000
# Remove update variable and save environment
setenv updatedev
saveenv
# Done
echo "Installation complete"
echo
echo "Please set/verify ethernet address(es) now and call saveenv"
```

Figure 2: Script update.txt

The script can be converted to the u-boot script by mkimage tool from our bsp. This can be done with the following command:

```
mkimage -A arm64 -O u-boot -T script C none -n "F&S update script" -d update.txt update.scr
```

For more details see section 6.4.7 of the “Linux on F&S Boards” documentation.

Important Notice

The information in this publication has been carefully checked and is believed to be entirely accurate at the time of publication. F&S Elektronik Systeme assumes no responsibility, however, for possible errors or omissions, or for any consequences resulting from the use of the information contained in this documentation.

F&S Elektronik Systeme reserves the right to make changes in its products or product specifications or product documentation with the intent to improve function or design at any time and without notice and is not required to update this documentation to reflect such changes.

F&S Elektronik Systeme makes no warranty or guarantee regarding the suitability of its products for any particular purpose, nor does F&S Elektronik Systeme assume any liability arising out of the documentation or use of any product and specifically disclaims any and all liability, including without limitation any consequential or incidental damages.

Products are not designed, intended, or authorised for use as components in systems intended for applications intended to support or sustain life, or for any other application in which the failure of the product from F&S Elektronik Systeme could create a situation where personal injury or death may occur. Should the Buyer purchase or use a F&S Elektronik Systeme product for any such unintended or unauthorised application, the Buyer shall indemnify and hold F&S Elektronik Systeme and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, expenses, and reasonable attorney fees arising out of, either directly or indirectly, any claim of personal injury or death that may be associated with such unintended or unauthorised use, even if such claim alleges that F&S Elektronik Systeme was negligent regarding the design or manufacture of said product.