

BFlash (v 7.31) User Manual

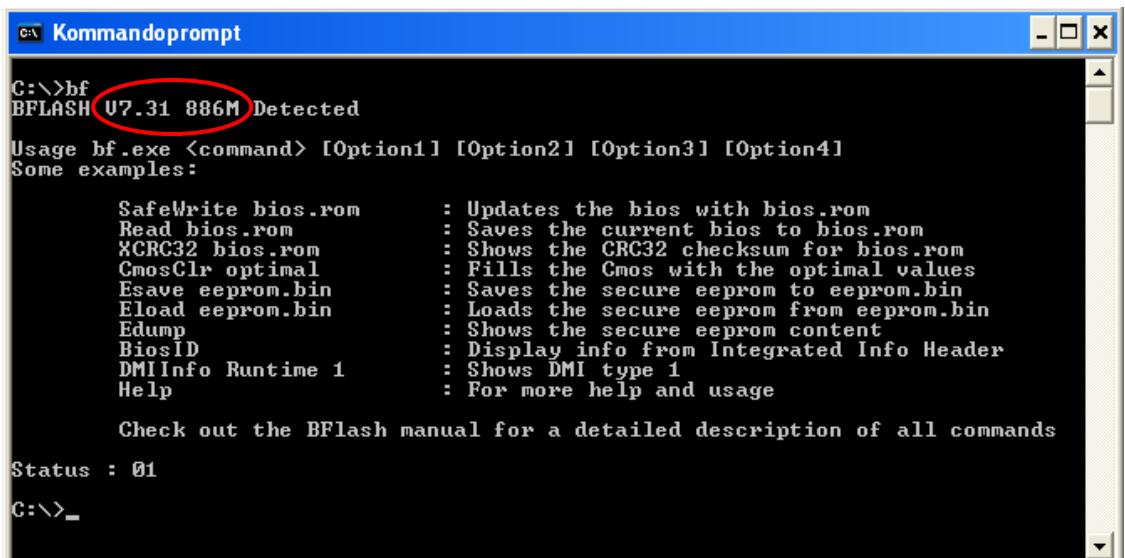
BFlash (BIOS Flash) is a software tool (BF.EXE) designed for BIOS and EEPROM operations on Kontron Embedded/Basic Motherboard families:

KTGV41, KTUS15, KT780, KT690, KT965, 986LCD-M, 886LCD-M, 886LCD-GV and 786LCD/mITX.

In this user manual the abbreviation BF is used for BFlash. This also match with the actual name of the SW tool (BF.EXE). BF is supported by DOS/Win2000/WinXP and can be used to read and write data to - and from BIOS Flash RAM, EEPROM and CMOS storage. By this tool it is possible to update BIOS, change DMI codes, setup Vendor codes and save copies of all data. The copies can be used as Master data for mass production.

Type BF <ret> from DOS prompt to see the BFlash version number and the actual board version.

As an example the version in the picture below is 7.31 and the actual board is 886M (886LCD-M).



```
C:\>bf
BFLASH V7.31 886M Detected

Usage bf.exe <command> [Option1] [Option2] [Option3] [Option4]
Some examples:

SafeWrite bios.rom      : Updates the bios with bios.rom
Read bios.rom           : Saves the current bios to bios.rom
XCRC32 bios.rom         : Shows the CRC32 checksum for bios.rom
CmosClr optimal         : Fills the Cmos with the optimal values
Esave eeprom.bin        : Saves the secure eeprom to eeprom.bin
Eload eeprom.bin        : Loads the secure eeprom from eeprom.bin
Edump                   : Shows the secure eeprom content
BiosID                  : Display info from Integrated Info Header
DMIInfo Runtime 1       : Shows DMI type 1
Help                    : For more help and usage

Check out the BFlash manual for a detailed description of all commands

Status : 01
C:\>_
```

Copyright Notice:

Copyright © 2007, KONTRON Technology A/S, ALL RIGHTS RESERVED.

No part of this document may be reproduced or transmitted in any form or by any means, electronically or mechanically, for any purpose, without the express written permission of KONTRON Technology A/S.

Disclaimer:

KONTRON Technology A/S reserves the right to make changes, without notice, to any product, including circuits and/or software described or contained in this manual in order to improve design and/or performance. Specifications listed in this manual are subject to change without notice. KONTRON Technology assumes no responsibility or liability for the use of the described product(s), conveys no license or title under any patent, copyright, or mask work rights to these products, and makes no representations or warranties that these products are free from patent, copyright, or mask work right infringement, unless otherwise specified. Applications that are described in this manual are for illustration purposes only. KONTRON Technology A/S makes no representation or warranty that such application will be suitable for the specified use without further testing or modification.

Introduction

BF is used by typing BF and optionally a command and optionally followed by one or more parameters. This can be done from a DOS prompt or from a BAT file. The BAT file can be executed also from Windows.

The BF command line syntax is always: *bf <command> [Option1] ... [OptionN]*

Where <> is the name of the required command and [] is optional parameter(s)

The *BF*, *<command>*, *[Option1]* to *[OptionN]* are all separated by space character.

(If no command is entered then the response will be as shown on previous page).

The *[Option1]* to *[OptionN]* can be a sub-command, a file name, an address, an address space, data bytes/Words/DWords, text string and a count number.

Example: *bf safewrite bios.rom*

This command line will make a safe write of the bios.rom file. More specific it will be verify that the parameter bios.rom is a BIOS file and that it matches the actual board and further more that the file is not corrupted. This command line requires that bf.exe and bios.rom is located in the same directory.

A BAT file can contain one or more BF command lines.

Example: *RdMast10.bat* can have the content:

```
@ECHO OFF
```

```
echo This will generate Master files of the actual BIOS (size 1MB) and CMOS settings.
```

```
echo *** Make sure "Secure CMOS" = Enabled or "OEM Failsafe default" is active ***
```

```
pause
```

```
bf esave Mastprom.bin
```

```
echo MasterROM is generated (Secure CMOS values are dumped to Mastprom.bin)
```

```
bf read MastBIOS.rom 0 100000
```

```
echo MasterBIOS is generated (BIOS dumped to MastBIOS.rom)
```

```
echo Reading is finished.
```



Warning: Incorrect use of BF can corrupt BIOS and/or EEPROM and make the board unstable or even not boot anymore. It's recommended to use the command "SafeWrite" instead of "Write" to make sure not to load wrong or corrupted BIOS types.

Windows Vista/2008 Note:

In case BF is executed from Windows Vista/2008 you have to disabled UAC (see below how to do) before BF will work. If UAC is not disabled you will see the error message "OpenSCManager Failed, Access Denied."

To disable UAC:

- Start "msconfig" from a command prompt

- Goto "Tools"

- Find "Disable UAC" Option and select it

- Reboot system.

Command syntaxes

Below is a list of all the BF command syntaxes. The most common commands are marked in **bold**.

BiosID	
CmosClr	[Optimal/Failsafe]
CmosDump	
CmosLoad	<FileName>
CmosSave	<FileName>
CmosWrite	<Flash Address> <Data> [Data]
DMIInfo	<Runtime/Onboard/Romfile> [File Name/Dmi Type]
DMIWrite	<DMIScript> <FileName/Onboard>
DMISave	<DMIScript>
Dump	<Flash address>
ECIr	
EDump	[Memory Address] [Dump Length]
Eload	<FileName.dat>
EPoke	<EEPROM Address> <Data> [Data]
ESave	<FileName.dat>
FlashID	
FlashSupport	
Help	
MDump	<Memory Address> [Dump Length]
MDumpd	<Memory Address> [Dump Length]
MDumpw	<Memory Address> [Dump Length]
MPeek	<Mem Address>
MPoke	<Memory Address> <Data> [Data]
MPoked	<Memory Address> <Data> [Data]
MPokew	<Memory Address> <Data> [Data]
Mread	<Filename> <Address> <Count>
Mwrite	<Filename> <Address>
Peek	<Flash address>
Poke	<Flash address> <Data>
Read	<File name> <Flash address> <Count>
SafeWrite	<Filename> [CRC32]
SLP	[SLPKey/TextFile] [Data][Romfile/Onboard]
Write	<File name> <Flash address> [Exclude xxxxxx-xxxxxx]
XCRC32	<FileName>

Command descriptions

The most common commands are marked in **bold**.

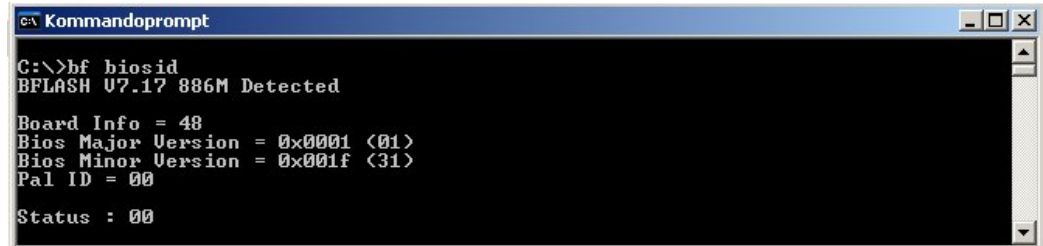
BiosID

(This command has no parameter)

Ex. "BF BiosID"

Display board Info no., BIOS Major and Minor numbers, PAL ID no.

Example:



```

C:\>bf biosid
BFLASH V7.17 886M Detected

Board Info = 48
Bios Major Version = 0x0001 <01>
Bios Minor Version = 0x001f <31>
Pal ID = 00

Status : 00
  
```

The board info number is 48, so from the list below it means that the actual board is an 886LCD-M/mITX. BIOS version is 1.31 which is displayed in BIOS as the BIOS ID = 886LCD31 and the Pal ID is displayed in BIOS as PCB ID = 00.

Board Info list:

0 = 486LCD	(EOL)
1 = 586LCD	(EOL)
2 = 686LCD/S	(EOL)
3 = 686LCD/MG	(EOL)
4 = 586GXM	(EOL)
5 = 686GXM	(EOL)
6 = GXMBASIC	(EOL)
10 = GX1LCD/S	(EOL)
11 = GX1LCD/S Plus	(EOL)
18 = GX1LCD/3"5 Standard	(EOL)
19 = GX1LCD/3"5 Plus	(EOL)
20 = 786LCD/S	(EOL)
21 = 786LCD/MG	(EOL)
24 = 786LCD/ST	(EOL)
28 = 786LCD/5.25	(EOL)
30 = ETXplus-786	(EOL)
40 = 886LCD-M/Flex	
48 = 886LCD-M/mITX	
50 = 886LCD-M/ATX	
60 = 786LCD/mITX	
90 = 986LCD/mITX	
91 = 986LCD-M/Flex	
92 = 986LCD-M/ATX	
93 = 986LCD-M/ATXE	
94 = KT965/Flex	
95 = KT965/ATXE	
96 = KT965/ATXP	

CmosClr

[Optimal/Failsafe]

CMOS memory will be cleared and optionally loaded with Optimal - or Failsafe values.

Ex. "BF CmosClr Optimal"

This will load Optimal values into CMOS memory.

Ex. "BF CmosClr"

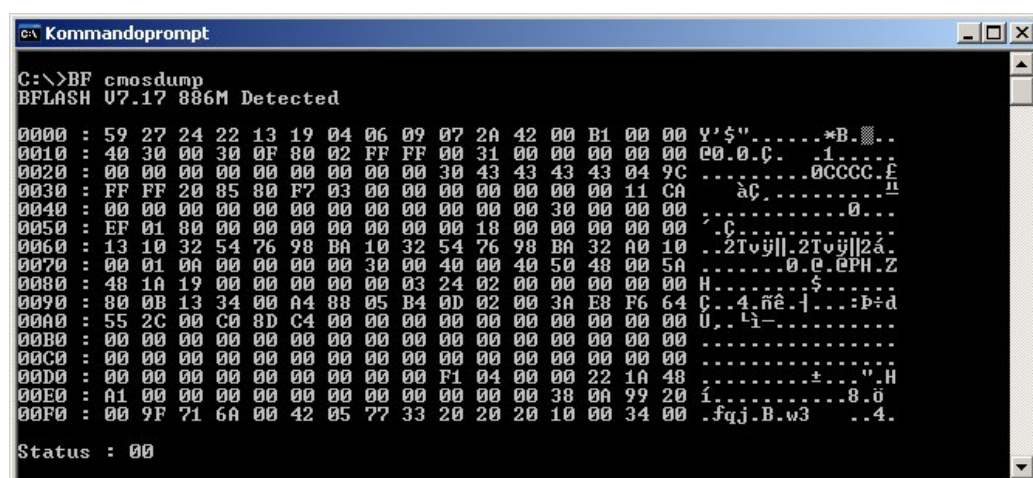
This will fill the whole CMOS memory with 0x00

CmosDump

(This command has no parameter)

Ex. "BF CmosDump"

Displays the CMOS data



```

C:\>BF cmosdump
BFLASH U7.17 886M Detected
0000 : 59 27 24 22 13 19 04 06 09 07 2A 42 00 B1 00 00 Y'$. . . . .B. . .
0010 : 40 30 00 30 0F 80 02 FF FF 00 31 00 00 00 00 00 e0.0.ç. .1. . .
0020 : 00 00 00 00 00 00 00 00 00 00 30 43 43 43 04 9C . . . . .0CCCC.f
0030 : FF FF 20 85 80 F7 03 00 00 00 00 00 00 00 11 CA . . . . .àç. . . . .u
0040 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 . . . . .0. . . .
0050 : EF 01 80 00 00 00 00 00 00 00 18 00 00 00 00 00 .ç. . . . .
0060 : 13 10 32 54 76 98 BA 10 32 54 76 98 BA 32 A0 10 .2Tvy||.2Tvy||2á.
0070 : 00 01 0A 00 00 00 00 00 30 00 40 00 40 50 48 5A . . . . .0.e.ePH.Z
0080 : 48 1A 19 00 00 00 00 00 03 24 02 00 00 00 00 H. . . . .$.
0090 : 80 0B 13 34 00 A4 88 05 B4 0D 02 00 3A E8 F6 64 ç.4.ñê.}. . . :p÷d
00A0 : 55 2C 00 C0 8D C4 00 00 00 00 00 00 00 00 00 U. .i- . . . .
00B0 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 . . . . .
00C0 : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 . . . . .
00D0 : 00 00 00 00 00 00 00 00 00 00 F1 04 00 22 1A 48 . . . . .±. . . .H
00E0 : A1 00 00 00 00 00 00 00 00 00 00 00 38 0A 99 20 í. . . . .8.ö
00F0 : 00 9F 71 6A 00 42 05 77 33 20 20 20 10 00 34 00 .fqj.B.w3 . .4.

Status : 00
  
```

CmosLoad

<File name>

File containing CMOS data to be loaded

Ex. "BF CmosLoad cmos.bin"

Copy 256 bytes from the cmos.bin file, except the first 16 bytes which contains date & time and some status bits.

CmosSave

<File name>

File where the CMOS data will be saved

Ex. "BF CmosSave cmos.bin"

Saves the content of the CMOS RAM to a binary file.

CmosWrite

<Flash Address>

Start address for data to be written

<Data>

Data to be written to CMOS RAM

[data]

Optionally more data

Ex. "BF CmosWrite c0 08 09 0A 0B"

This will write 4 bytes to the CMOS starting at address 0xC0.

DMIIInfo

<Runtime/Onboard/Romfile>

Runtime: Read DMI from running system. This is what you will see in Windows ex.

Onboard: Read DMI from onboard Bios rom

Romfile: Read DMI from a Bios rom file

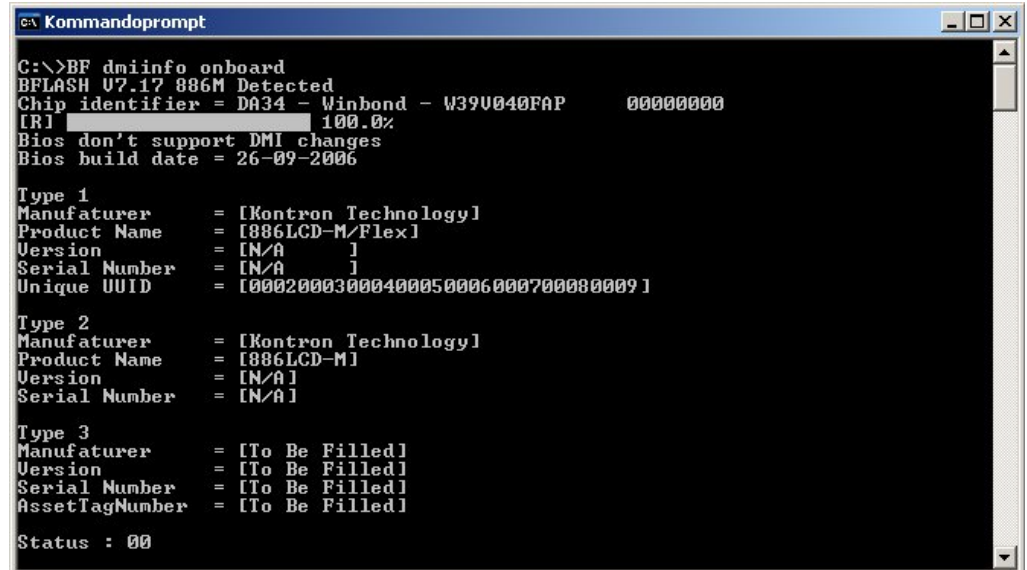
[FileName/DmiType]

FileName: used for Romfile option.

Dmi type: Optional use for Runtime or Onboard option, format is integer 0 – 127.

Ex. “BF DMIInfo onboard”

This will show DMI info from the onboard BIOS (not runtime BIOS), see picture below.



```

C:\>BF dmiinfo onboard
BFLASH U7.17 886M Detected
Chip identifier = DA34 - Winbond - W39U040FAP 00000000
[RI] 100.0%
Bios don't support DMI changes
Bios build date = 26-09-2006

Type 1
Manufacturer = [Kontron Technology]
Product Name = [886LCD-M/Flex]
Version = [N/A]
Serial Number = [N/A]
Unique UUID = [00020003000400050006000700080009]

Type 2
Manufacturer = [Kontron Technology]
Product Name = [886LCD-M]
Version = [N/A]
Serial Number = [N/A]

Type 3
Manufacturer = [To Be Filled]
Version = [To Be Filled]
Serial Number = [To Be Filled]
AssetTagNumber = [To Be Filled]

Status : 00
  
```

Note: If “N/A” then value will be loaded from EPROM in the runtime BIOS.

DMI Type 1 – 3 configuration table:

DMI type	String name	Max. no of char.	Type	Default string content
1	'Manufacturer='	18	ASCII	'Kontron Technology'
	'ProductName='	13	ASCII	'886LCD-M/Flex'
	'Version='	8	ASCII	'N/A'
	'SerialNumber='	8	ASCII	'N/A'
	'UniqueUUID='	32	Hex	'0123456789ABCDEF0123456789ABCDEF'
2	'Manufacturer='	18	ASCII	'Kontron Technology'
	'ProductName='	13	ASCII	'886LCD-M/Flex'
	'Version='	8	ASCII	'N/A'
	'SerialNumber='	8	ASCII	'N/A'
3	'Manufacturer='	12	ASCII	'To Be Filled'
	'Version='	12	ASCII	'To Be Filled'
	'SerialNumber='	12	ASCII	'To Be Filled'
	'TagNumber='	12	ASCII	'To Be Filled'

☐ = Not possible to setup by OEM (DMI Type 2 is reserved by Kontron)

Ex. “BF DMIInfo romfile bios.rom”

This will show DMI info from a BIOS file. The output will be identical to picture above if onboard BIOS and bios.rom file are the same, the only difference is that BF will not display “Chip identifier” information.

Ex. "BF DMIInfo runtime 1"

This will show DMI info type 1 from memory (runtime BIOS), see picture below.



```

c:\>bf dmiinfo runtime 1
BFLASH U7.17 886M Detected

DMI Info Version 2.3 Found with 52 structures

System Information <Type 01 Handle 0001>
Manufacturer: Kontron Technology
Product Name: 886LCD-M/mITX
Version: 53810000
Serial Number: 00300279
UUID: 00020003-0004-0005-0006-000700080009
Wake-up Type: Power Switch

Status : 00
  
```

DMISave <FileName> File (ScriptFile) to contain info about DMI type 1 & 3

Ex. "BF DMISave script.txt"

This will copy DMI Type 1 and Type 3 (located in BIOS) to the script.txt file.
(Example of script.txt file is in the DMIWrite description).

DMIWrite <DMIScript> A DMI script file containing info about DMI type 1 & 3
<FileName/Onboard> Filename: Write DMI data to (BIOS rom) file
Onboard: Write DMI data to onboard BIOS

Ex. "BF DMIWrite script.txt onboard"

This will do the changes specified in the script.txt file to the current onboard bios.

Ex.1 Script-file (all settings):

```

'Type=' '1'
'ProductName=' '886LCD-M/Flex'
'SerialNumber=' '11012345'
'UniqueUUID=' '0123456789ABCDEF0123456789ABCDEF'
'Type=' '3'
'Manufacturer=' 'Kontron America'
'Version=' '886LCD-M/Flex'
'SerialNumber=' '1050AABB'
'TagNumber=' '50012345'
  
```

Note: Ping ' (ASCII no. 27h) is used as separator and can't be used in DMI text strings.
'Type=' '1' and 'Type=' '3' must always be present, other parameters are optional. All optional parameters (except UniqueUUID) can have the value '?' to manually enter value via keyboard. The UniqueUUID must always contain 32 hex characters, but if last character is '?' then the last 8 characters shall be entered via keyboard.

Dump <Flash address> Start address for reading data from Flash RAM

Ex. "BF Dump 007F00"

This will display the content of the Flash address 0x007F00 and 100 bytes ahead.

ECIr (This command has no parameter)

Ex. "BF ECIr"

This will clear the EEPROM, (all bytes = 00h).

EDump [EEPROM Address] Start address to read data
[Data Length] Number of bytes to read (in Hex)

Ex. "BF EDump E0 0F"

This will display 16 bytes from address 0xE0.

ELoad <FileName> File containing EEPROM data

Ex. "BF ELoad eeprom.bin"

This will copy the eeprom.bin file content to the EEPROM

EPoke <EEPROM Address> Start address for writing data
<Data> Data to be written to the EEPROM
[Data] Optionally more data

Ex. "BF EPoke F0 12 34 56 78"

This will 4 bytes to the EEPROM starting from address 0xF0.

ESave <FileName> File to contain EEPROM data

Ex. "BF ESave eeprom.bin"

This will save the content of the EEPROM to eeprom.bin.

FlashID (This command has no parameter)

Ex. "BF FlashID"

This will display the type of flash detected on the flash module.

FlashSupport (This command has no parameter)

Ex. "BF FlashSupport"

Shows the different Flash RAM chip types supported by BFlash.

Help (This command has no parameter)

Ex. "BF Help"

Shows help about BFlash most common commands.

MDump <Memory Address> Start address where to read data
[Dump length] Number of bytes to read

Ex. "BF MDump F0000 0F"

This displays 16 bytes starting from memory address 0xF0000. (Windows can crash).

MDumpD <Memory Address> Start address where to read DWords data
[Dump length] Number of bytes to read

Ex. "BF MDumpD F0000 0F"

This displays 4 DWords starting from memory address 0xF0000. (Windows can crash).

MDumpW <Memory Address> Start address where to read Words data
 [Dump length] Number of bytes to read

Ex. "BF MDumpW F0000 0F"

This displays 8 Words starting from memory address 0xF0000. (Windows can crash).

MPeek <Memory Address> Address containing byte to read

Ex. "BF MPeek F00000"

This will display a byte from memory address 0xF00000.

MPoke <Memory Address> Start address where to write byte data
 <Data> Byte data to be written to memory
 [data] Optionally more data

Ex. "BF MPoke F00000 FF"

This will write FFh to the memory at address 0xF00000.

MPokeD <Memory Address> Start address where to write DWord data
 <Data> DWord data to be written to memory
 [data] Optionally more data

Ex. "BF MPokeD F00000 12345678"

This will write a Dword to the memory at address 0xF00000.

MPokeW <Memory Address> Start address where to write Word data
 <Data> Word data to be written to memory
 [data] Optionally more data

Ex. "BF MPokeW F00000 1234"

This will write a word to the memory at address 0xF00000.

MRead <Filename> File to contain copy of memory data
 <Memory Address> Start address for reading
 <Count> Number of bytes to read (Hex format).

Ex. "BF MRead memory.bin F00000 20"

This will save 2x16 bytes of memory from address 0xF00000 to the file memory.bin.

MWrite <Filename> File containing data to be written to memory
 <Memory Address> Start address for writing data

Ex. "BF MWrite memory.bin F00000"

This will write the memory.bin to memory address 0xF00000.

Peek <Flash address> Flash address to read a byte

Ex. "BF Peek 007F10"

This will display the byte at location 0x007F10 and save it in "DOS ERRORLEVEL".

Poke <Flash address> Flash address where to write a byte
<Data> Byte data to write

Ex. "BF Poke 007F10 00"

This will store 00 at location 007F10 in the Flash.

Read <File name> File to contain Flash data
<Flash address> Start location on the Flash
<Count> Number of bytes to copy to file

Ex. "BF Read dump.rom 000000 008000"

Copies 008000 bytes to dump.rom starting at address 0x000000.

SafeWrite <File name> File containing Flash data (bios.rom file)
[CRC32] CRC32 is an 8hex char. value which can be generated by the XCRC32 command on the requested bios.rom file

Ex. "BF SafeWrite bios.rom 9abc1029"

If the Bios.rom matches the actual board type and if CRC32 checksum matches the 8 character value then Bios.rom will be burned into the actual BIOS Flash RAM.

SLP [SLPKey/TextFile] SLPKey: Up to 32 data bytes to follow
TextFile: File containing max. 32 bytes
(Space characters are allowed in the txt-file)
[Data] Optionally data to be used (SLPKey only)
(Space character not allowed in Data)
[RomFile/Onboard] RomFile: For copy of onboard BIOS w. SLP key
Onboard: SLP key will be saved in Bios

Ex. "BF SLP SLPKey KontronTechnology onboard"

This will copy the SLP key "KontronTechnology" into the onboard BIOS.
(If space must be used in the key then use the TextFile command).

Ex. "BF SLP TextFile oemkey.txt onboard"

This will copy the SLP key located in oemkey.txt into the onboard BIOS.

Ex. "BF SLP SLPKey XpOEMString bios.rom"

This will copy the string "XpOEMString" into the onboard bios SLP area and then save a copy of the onboard bios (with SLP key) into a file called bios.rom.

Notes: SLP = **S**ystem **L**ocked **P**reinstallation.
"bf mdump fe840 1f" will display the SLP code.
(fe840 = F000:E840)

Write

<File name>

File containing BIOS to burn into Flash

<Flash address>

Start address of the Flash

[Exclude xxxxxx-xxxxxx]

Prevent overwriting data in Flash address area

Ex. "BF Write bios.rom 0"

This will burn the file bios.rom to address 0 in the flash.

XCRC32

<File name>

Input file for checksum calculation

Ex. "BF XCRC32 bios.rom"

This will calculate the 32bit checksum on the bios.rom file.

Appendix A: Command syntaxes (EOL products)

EOL legacy products:

786LCD family: 786LCD/S, 786LCD/S MG, 786LCD/ST, 786LCD/3'5 ST
 GX1LCD family: GX1LCD/S, GX1LCD/3'5
 and GXM -, 686LCD – and 486LCD family

Command syntaxes:

BIOSUpdate <File Name>
 Download <File name> [Both/Master]
 Erase [Master/Slave/Both]
 MakeRecover
 PalUpdate
 Recover
 Test [Both/Master]
 Upload <File name> [Both/Master]
 XErase [Master/Slave/Both]

Command descriptions

BiosUpdate <File name> File to be burned
 (786LCD only)

Ex. “BF BiosUpdate bios.rom”

Updates the bios with bios.rom.

Download <File name> File where to save data from Flash Disk
 (486LCD, [Both/Master] Default is slave Flash Disk module
 686LCD only) Master is primary Flash Disk module
 Both to select both Flash Disk modules

Ex. “BF Download master.img Both”

This will download all the data from both flash modules.

Erase [Master/Slave/Both] This is for selecting the flash module
 (486LCD, Default is slave (secondary module)
 686LCD only) Master will select primary module
 Both will select both modules

Ex. “BF Erase Both”

This will erase both the primary and secondary modules inclusive BIOS.

Warning: New bios must be burned before reset otherwise system can't boot anymore.

MakeRecover (This command has no parameter)
 (486LCD,
 686LCD only)

Ex. “BF MakeRecover”

This will copy the Master flash to the Slave flash

PalUpdate (This command has no parameter)
 (786LCD-ST only)

Ex. “BF PalUpdate”

Updates the PAL

Recover

(486LCD,
686LCD only)

(This command has no parameter)

Ex. "BF Recover"

This will copy the Slave flash to the Master flash

Test

(486LCD,
686LCD only)

[Both/Master]

This is for selecting the flash module
Default is slave (secondary module)
Master will select primary module
Both will select both modules

Ex. "BF Test Both"

This will erase both the primary and secondary modules inclusive BIOS.

Warning: New bios must be burned before reset otherwise system can't boot anymore.

Upload

(486LCD,
686LCD only)

<File name>
[Both/Master]

File containing image data for FD (Flash Disk)
Default is slave (secondary module)
Master: Select primary module as target FD
Both: Select both modules as target FD

Ex. "BF Upload master.img Both"

This will upload the file called master.img into both Flash Disk modules.

XErase

(486LCD,
686LCD only)

[Master/Slave/Both]

Flash module(s) to be erased

Ex. "BF XErase Both"

Erases both Flash modules

Appendix B: Examples of using BF in BAT-files

ClrOEM.bat (Remove OEM Failsafe)

```
@echo off
echo clroem.bat (Remove OEM Failsafe)
echo This will clear the OEM defaults (Secure CMOS setting will be visible in BIOS)
pause
bf epoke 0d 4c
```

RdMast05.bat (Make set of Master Files, BIOS size 0.5 MB)

```
@echo off
echo RdMast05.bat (Make set of Master Files, BIOS size 512KB)
echo This will generate Master files of the actual BIOS (size 512KB) and CMOS settings.
echo *** Make sure "Secure CMOS" = Enabled or "OEM Failsafe default" is active ***
pause
bf esave mastprom.bin
echo MasterPROM is generated (Secure CMOS values are dumped to mastprom.bin)
bf read mastbios.rom 0 80000
echo MasterBIOS is generated (BIOS dumped to mastbios.rom)
echo Reading is finished.
```

RdMast10.bat (Make set of Master Files, BIOS size 1MB)

```
@echo off
echo RdMast10.bat (Make set of Master Files, BIOS size 1024KB)
echo This will generate Master files of the actual BIOS (size 1024KB) and CMOS settings.
echo *** Make sure "Secure CMOS" = Enabled or "OEM Failsafe default" is active ***
pause
bf esave mastprom.bin
echo MasterPROM is generated (Secure CMOS values are dumped to mastprom.bin)
bf read mastbios.rom 0 100000
echo MasterBIOS is generated (BIOS dumped to mastbios.rom)
echo Reading is finished.
```

SetOEM.bat (Set OEM Failsafe)

```
@echo off
echo SetOEM.bat (Set OEM Failsafe settings)
echo This will set the OEM defaults (Secure CMOS setting will be invisible in BIOS)
pause
bf epoke 0d 00
```

TypeDMI.bat (Manually type in DMI codes)

```
@echo off
echo Manually enter Type 3 Serial number and Tagnumber
pause
bf DMIWrite UniqDMI.txt onboard
```

The file UniqDMI.txt contains:

'Type='	'1'
'Type='	'3'
'SerialNumber='	'?'
'TagNumber='	'?'

Upd.bat (Update standard BIOS)

```
@echo off
echo Upd.bat (Update standard BIOS)
echo This will burn bios.rom file and fill cmos with optimal values
pause
bf safewrite bios.rom
bf cmosclr optimal
```

UpdOnDMI.bat (Update BIOS and keep DMI)

```
@echo off
echo UpdOnDMI.bat (Update BIOS and keep DMI)
echo This will burn bios.rom file and fill CMOS with optimal values
echo but DMI code will remain
pause
bf dmisave script.txt
bf safewrite bios.rom
bf dmiwrite script.txt onboard
bf cmosclr optimal
```

WrDMI.bat (Manually type in DMI codes)

```
@echo off
echo Write Type 1 and Type 3 DMI settings via script file
pause
bf DMIWrite Script.txt onboard
```

The file Script.txt contains:

'Type='	'1'
'ProductName='	'886LCD-M/Flex'
'SerialNumber='	'11012345'
'UniqueUUID='	'0123456789ABCDEF0123456789ABCDEF'
'Type='	'3'
'Manufacturer='	'Kontron America'
'Version='	'886LCD-M/Flex'
'SerialNumber='	'1050AABB'
'TagNumber='	'50012345'

WrMast.bat (Write Master Files)

```
@echo off
echo WrMast.bat (Write Master Files)
echo This will copy the BIOS and CMOS Master file to the actual board.
echo Note the "Secure CMOS" gets enabled or OEM Failsafe defaults gets active.
pause
bf safewrite mastbios.rom
bf eload mastprom.bin
```