TEXAS INSTRUMENTS

User's Guide

SLAU654E–November 2015–Revised February 2019

MSP Flasher

MSP Flasher is a user-friendly shell-based interface that provides easy access to MSP devices through JTAG or Spy-Bi-Wire (SBW) by porting the most common functions of the MSP Debug Stack to the command line.

Contents

List of Figures

List of Tables

Trademarks

eZ430, LaunchPad, eZ430-Chronos, MSP430, MSP432, SimpleLink are trademarks of Texas Instruments.
OS X is a registered trademark of Apple Inc.
Ubuntu is a registered trademark of Canonical Ltd.
Windows is a registered trademark of Microsoft Corporation.
All other trademarks are the property of their respective owners.

1 Introduction

The typical MSP Flasher execution flow consists of the following steps. Optional steps can be activated or deactivated by using special triggers or parameters (see Section 3).

1. Initialize FET debugger
2. Perform FET recovery (if a corrupted FET firmware is detected)
3. Update FET firmware (if a mismatch between firmware and MSP Debug Stack versions is detected)
4. Power up the target MSP device
5. Configure the target MSP for JTAG or SBW communication
6. Connect to the target MSP and display device information
7. Optional: Erase (parts of) the target device memory
8. Optional: Load target code into the device from a TXT or HEX file
9. Optional: Verify target code transfer
10. Optional: Read device memory and write it to a TXT or HEX file
11. Optional: Reset the device
12. Optional: Lock JTAG access
13. Optional: Reset the device
14. Optional: Power down the device
15. Optional: Start target code execution
16. Disconnect from the target MSP device
17. Close the FET connection

Status reports are written to a text file named log.txt. This file is saved in the Log folder under the folder where the MSP Flasher executable resides. If the Log folder does not exist, it is automatically created. New instances are appended to the log file, and old logs are never overwritten.

NOTE: For a GUI-based alternative to MSP Flasher, see UniFlash. As of version 4.0, UniFlash features a command line interface with MSP Flasher compatibility mode.

2 Compatibility

MSP Flasher supports the following operating systems:

- Windows® 7 32 bit or 64 bit
- Windows 8 32 bit or 64 bit
- Windows 10 32 bit or 64 bit
- Ubuntu® 12.04 32 bit or 64 bit
- Ubuntu 14.04 32 bit or 64 bit
- Ubuntu 16.04 32 bit or 64 bit
- OS X® 10.9 or newer

NOTE: **MSP Flasher for Linux does not support eZ430™ development tools.** This includes the Value Line LaunchPad™ development kit with eZ430 onboard emulation, eZ430-Chronos™ development tool, and older MSP-EXP430 experimenter boards with eZ430 onboard emulation.

MSP Flasher requires a hardware interface to communicate with MSP target devices. The following TI flash emulation tools (FETs) are supported:

- MSP-FET
- MSP-FET430UIF
- eZ-FET and eZ-FET lite
- eZ430 (including LaunchPad development kits)

NOTE: **Do not disconnect the JTAG or emulator USB cable while MSP Flasher is running.** Wait until MSP Flasher execution is finished before disconnecting the debugger or target device.

NOTE: To differentiate between multiple eZ430 tools (for example, two or more Value Line LaunchPad tools connected to the same host PC), connect each tool individually or use the unique identifier that is reported by MSP Flasher.
("Found USB FET @ **HID0xxx:COMxxx**").

Use this identifier with the **–I** switch whenever more than one eZ430 debugger is connected.

3 Triggers and Arguments

MSP Flasher runs from an executable file called MSP430Flasher. This file accepts a number of triggers and arguments to access the full capabilities of the software. Table 1 lists all available triggers and arguments.

Table 1. Available Triggers and Arguments[1]

Trigger	Arguments	Description and Additional Information
-h / -?	N/A	Displays usage information (displays this table of command line switches)
-x	N/A	Displays available exit specifications (see trigger -z)
-i	TIUSB **or** USB (**default**)	Communication port for the FET debugger. TIUSB (or USB) is the default. Use COM*n* (for example, COM15) on Windows or ttyACM*n* (for example, ttyACM15*)* on Linux or usbmodem*n* (for example, usbmodem1421) on OS X to choose a debugger connected to COM port *n.* Use HID*n*:COM*n* for specific eZ430 tools on Windows (see note in Section 2). Use **-i DETECT** to execute a FET detection sweep, to display detailed information about all connected debug tools, and to prompt to select a FET.
	COM*n* **or** ttyACM*n* or usbmodem*n*	
	HID*n*:COM*n*	
	DETECT	
-j	fast	Configures the MSP Debug Stack to increase or decrease the JTAG or SBW frequency of the FET.
	medium (**default**)	
	slow	
-n	Device name	Optional for MSP430™ MCUs, mandatory for MSP432™ MCUs. The name of the device being accessed (prompt if mismatch between found and selected device). **-n NO_TARGET** executes MSP Flasher without attempting to connect to a target device. Choose this option to detect if a certain FET is connected or when the FET firmware should be updated only.
	NO_TARGET	
-r	[Filename, mem_section]	Triggers a read operation in target device memory section specified by *mem_section.* The memory content is written to a file specified by *Filename.* Available memory sections are: MAIN = the main memory of the device INFO = info memory (see trigger –u) BSL = bootloader memory (see trigger –b) RAM = random access memory *0x****-0x***** = custom memory section Specify .txt as the extension for *Filename* to write data in TI-TXT format, or specify .a43 (or .hex) to write data in Intel-Hex format.

[1] Omitted mandatory arguments are replaced by the default options if possible, or the user is prompted to provide them later.

Table 1. Available Triggers and Arguments[1] (continued)

Trigger	Arguments	Description and Additional Information
-w	Filename	Triggers a memory write operation. The accepted formats are TXT (TI-txt) or HEX (Intel-hex).
-v	filename (optional)	Triggers verification of the target memory against a target code file. If -w is used, no argument is required. For a stand-alone verify, provide the path to a target code file as an argument.
-u	N/A	Unlocks locked flash memory (INFOA) for writing.
-b	N/A	Unlocks the BSL memory for writing.
-e	ERASE_ALL (**default**)	Triggers an erasure of the device's MAIN memory (ERASE_MAIN) or MAIN and INFO memory including the INFOA segment if unlocked (ERASE_ALL).
	ERASE_MAIN	
	ERASE_SEGMENT	See Section 6. Use only with the -w switch.
	ERASE_TOTAL	Triggers a complete erase of the target device memory, which overrides and resets any memory protection settings. Use this command for SimpleLink™ MSP432 devices to force a factory reset. This will avoid the pop up if active JTAG/SWD lock is detected.
	ERASE_USER_CODE	**Applicable for FR4xx devices only.** Overrides and clears FRAM memory protection (also see the *MSP430FR4xx and MSP430FR2xx family user's guide*) and erases INFO and MAIN memory.
	NO_ERASE	Target memory is not erased prior to programming. **Caution:** Overwriting previously programmed memory section without prior erase might result in data corruption on devices with flash memory. Use only with –w switch.
-p	JTAG password	Specifies the JTAG password that should be used to open a password protected target device (supported on FRAM devices only). The user is prompted if the password is incompatible with the password length specified by trigger -l.

-o	L	Operating mode for L092 and RF430FR152H family devices.
		L = L092 mode (normal mode)
		C = C092 mode (ROM development mode)

	C	
-f	N/A	Permanently secures JTAG access to the target MSP. **Caution:** The device will no longer be accessible through JTAG or Spy-Bi-Wire. This action is irreversible.
-g	N/A	Disables the logging mechanism.
-a	N/A	Causes a nonintrusive target connection: use this switch if no reset should be applied to the target device on start up. Correct target device name must be specified using the -n switch.
-s	N/A	Suppresses the FET firmware update user prompt. In case of a mismatch between MSP Debug Stack and FET firmware, an update is forced.
-q	N/A	QUIET mode. No system messages will be displayed (except for errors and user prompts).
-z	[exit_spec,...]	Specifies the state of the device after programming. For available exit specifications, see Table 2. Use "," as a delimiter.
-m	AUTO (**default**) SBW2, SBW4, JTAG	DEPRECATED. The applicable JTAG protocol is automatically detected by MSP Flasher. This trigger is ignored.
-l	password_length	DEPRECATED. The JTAG password length is automatically detected by MSP Flasher. This trigger is ignored.
-d	[breakpoint addresses]	DEPRECATED. The hardware breakpoint functionality is no longer maintained and will be removed in a future release of MSP Flasher.
-t	Timeout_in_ms	DEPRECATED. The hardware breakpoint functionality is no longer maintained and will be removed in a future release of MSP Flasher.

4 Exit Specifications

Select the desired state for the device to be set to when MSP Flasher finishes its operation. This can be done using the trigger -z and passing the arguments [exit_spec,…], where exit_spec is a valid exit specification shown in Table 2.

NOTE: The specifications are delimited with the ',' (comma) character and enclosed by square brackets.

Table 2. Available (Combinations of) Exit Specifications

Exit Specification	Description
default (-z not used)	The device does not receive a 'hard' reset and is powered down after programming. Target code execution does not start.
-z [VCC]	V_{CC} is set to the default value of 3000 mV. Target code execution starts.
-z [VCC=*3600*]	The target V_{CC} is set to a custom value (specified in millivolts). Valid voltages range from 1800 to 3600 mV. Target code execution starts. The eZ430 and eZ-FET debuggers do not support target voltages other than 3000 mV.
-z [RESET]	The device receives a 'hard' reset (using the $\overline{\text{RST}}$/NMI pin) after programming and is powered down.
-z [VCC(=*x*), RESET]-z [RESET, VCC(=*x*)]	The device receives a 'hard' reset (using the $\overline{\text{RST}}$/NMI pin) after programming and V_{CC} is left on. Target code execution starts.

5 Firmware Update

During runtime, if MSP Flasher detects a conflict between the firmware version of the debug probe (FET) and the version of the MSP Debug Stack (MSP430.dll), it prompts the user to let MSP Flasher update the firmware:

```
>> The firmware of your FET is outdated.
>> Would you like to update it? (Y/N): _
```

Type Y to update the firmware of the FET, display status reports, and on success continue execution of the MSP Flasher routine. Type N to resume the running instance with the outdated firmware. **TI recommends not using MSP Flasher while the FET firmware does not match the version of the MSP Debug Stack.**

If an error is detected during the update, MSP Flasher prompts the user to retry or cancel the update:

```
>> Update failed. (R)etry/(C)ancel? _
```

Type R to repeat the attempt to update. Type C to resume the running instance with the outdated firmware.

NOTE: The -s switch suppresses this user prompt. If there is a mismatch between the FET firmware version and the MSP Debug Stack version, a firmware update is done automatically.

NOTE: For fully automated FET firmware updates, run the following command:

```
MSP430Flasher -n NO_TARGET -s
```

MSP Flasher updates only the FET firmware and does not attempt to connect to any target MSP device.

6 Segment Erase

MSP Flasher supports erasure and reprogramming of a single memory segment while the rest of the device memory is left untouched. To use this feature, use the **-e** switch with the ERASE_SEGMENT option.

The user must provide a TI-txt or Intel-hex file that contains the target code in one continuous block. The start address of this memory block defines the segment that should be erased.

NOTE: The size of the memory block that to program must not exceed the size of the segment in which it should be programmed. Memory segments are either 256, 512, or 1024 bytes and have fixed addresses inside the main memory depending on the MSP430 device. Refer to the device user's guide and data sheet for the segment size and location for a specific target device.

NOTE: The entire segment will be erased prior to programming, even if the memory block to be programmed is smaller than the memory segment size.

It is also possible to leave the target memory unchanged before programming by using the -e NO_ERASE option. Thus, multiple memory blocks can be programmed into the device while leaving the memory sections in between them unchanged.

NOTE: Check the boundaries of the memory blocks to be programmed carefully when using the NO_ERASE option. Particularly on target devices with flash memory, writing without erasing can cause data corruption.

7 Example Cases

7.1 Loading and Executing Target Code From a TXT File

Details:

- Device: MSP430F5438A
- Interface: USB
- Password: N/A
- File: file.txt (in the same directory as the executable)
- Erase Type: ERASE_ALL
- Verification: TRUE
- VCC: ON

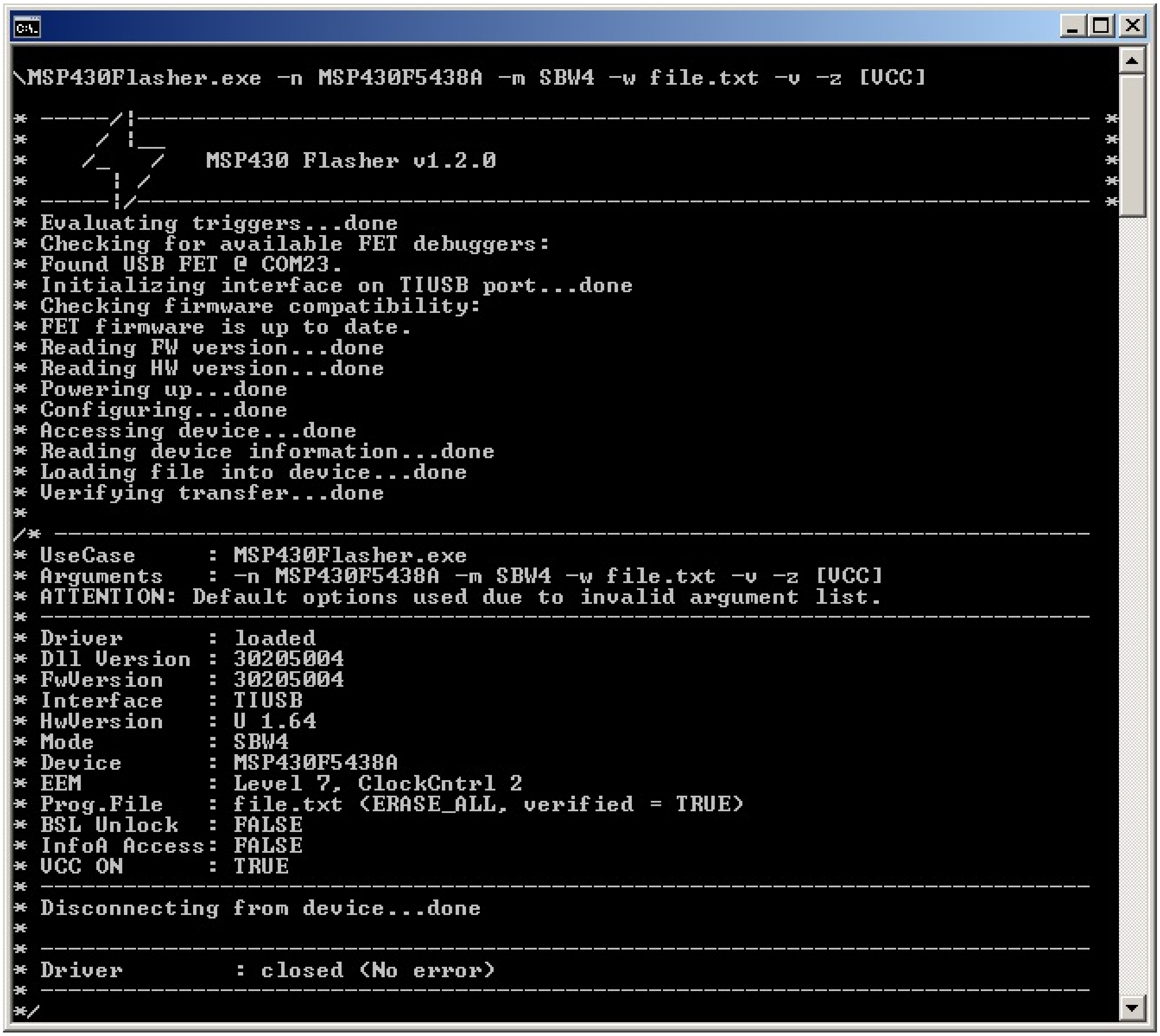
NOTE: To load a TI .txt or Intel .hex file, make sure that the file to be loaded is in the same directory as the executable or that a valid path is specified.

The command line to use in this case is:

```
MSP430Flasher -n MSP430F5438A -w file.txt -v -z [VCC] (-i USB) (-e ERASE_ALL)
```

NOTE: Triggers -p and -l are not used, because the device does not require a password. Triggers -i and -e may be used but are unnecessary, because USB and ERASE_ALL are the default settings for these parameters, respectively.

Figure 1 shows the console output on entering the previous command line into Windows command prompt if the selected device is connected through the specified COM port.



```
\MSP430Flasher.exe -n MSP430F5438A -m SBW4 -w file.txt -v -z [VCC]

* -----/|-------------------------------------------------------------------- *
*     / |__                                                                   *
*    /_   /   MSP430 Flasher v1.2.0                                           *
*      | /                                                                    *
* -----|/-------------------------------------------------------------------- *
* Evaluating triggers...done
* Checking for available FET debuggers:
* Found USB FET @ COM23.
* Initializing interface on TIUSB port...done
* Checking firmware compatibility:
* FET firmware is up to date.
* Reading FW version...done
* Reading HW version...done
* Powering up...done
* Configuring...done
* Accessing device...done
* Reading device information...done
* Loading file into device...done
* Verifying transfer...done
*
/* ----------------------------------------------------------------------------
* UseCase      : MSP430Flasher.exe
* Arguments    : -n MSP430F5438A -m SBW4 -w file.txt -v -z [VCC]
* ATTENTION: Default options used due to invalid argument list.
* -----------------------------------------------------------------------------
* Driver       : loaded
* Dll Version : 30205004
* FwVersion    : 30205004
* Interface    : TIUSB
* HwVersion    : U 1.64
* Mode         : SBW4
* Device       : MSP430F5438A
* EEM          : Level 7, ClockCntrl 2
* Prog.File    : file.txt (ERASE_ALL, verified = TRUE)
* BSL Unlock   : FALSE
* InfoA Access: FALSE
* VCC ON       : TRUE
* -----------------------------------------------------------------------------
* Disconnecting from device...done
*
* -----------------------------------------------------------------------------
* Driver         : closed (No error)
* -----------------------------------------------------------------------------
*/
```

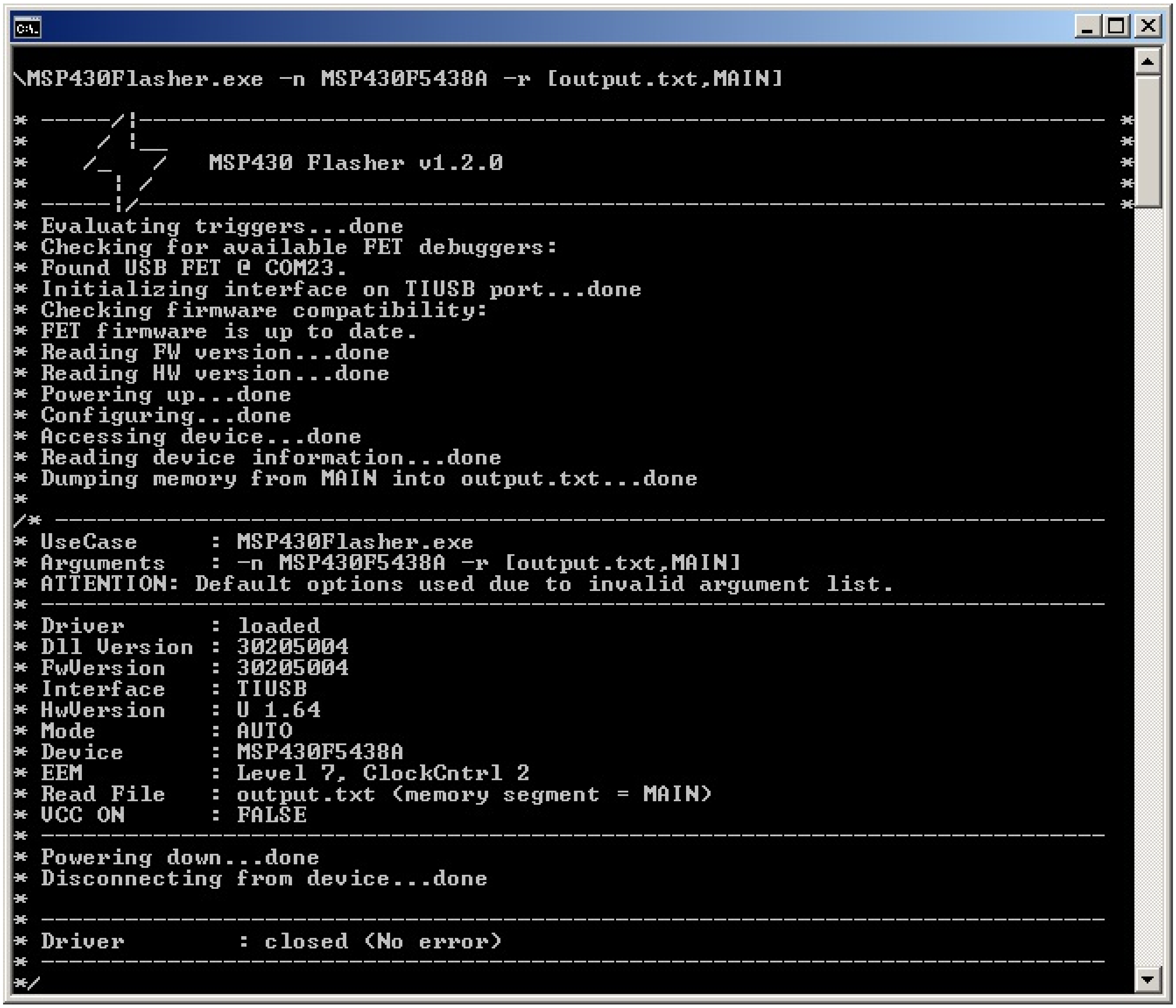
Figure 1. Loading and Executing Target Code From a .txt File

7.2 Reading Device Memory

MSP Flasher can read out any section of the device memory and write it to a file. The four memory sectors are MAIN, INFO, RAM, and BSL. In this example, the MAIN memory of an MSP430F5438A is written to a file named output.txt.

`MSP430Flasher -n MSP430F5438A -r [output.txt,MAIN]`

Figure 2 shows the console output after running the previous command line.



```
\MSP430Flasher.exe -n MSP430F5438A -r [output.txt,MAIN]

* -----/|---------------------------------------------------------------------- *
*     / |__                                                                     *
*    /_   /   MSP430 Flasher v1.2.0                                             *
*      | /                                                                      *
* -----|/----------------------------------------------------------------------- *
* Evaluating triggers...done
* Checking for available FET debuggers:
* Found USB FET @ COM23.
* Initializing interface on TIUSB port...done
* Checking firmware compatibility:
* FET firmware is up to date.
* Reading FW version...done
* Reading HW version...done
* Powering up...done
* Configuring...done
* Accessing device...done
* Reading device information...done
* Dumping memory from MAIN into output.txt...done
*
/* ----------------------------------------------------------------------------
* UseCase     : MSP430Flasher.exe
* Arguments   : -n MSP430F5438A -r [output.txt,MAIN]
* ATTENTION: Default options used due to invalid argument list.
* -----------------------------------------------------------------------------
* Driver      : loaded
* Dll Version : 30205004
* FwVersion   : 30205004
* Interface   : TIUSB
* HwVersion   : U 1.64
* Mode        : AUTO
* Device      : MSP430F5438A
* EEM         : Level 7, ClockCntrl 2
* Read File   : output.txt (memory segment = MAIN)
* VCC ON      : FALSE
* -----------------------------------------------------------------------------
* Powering down...done
* Disconnecting from device...done
*
* -----------------------------------------------------------------------------
* Driver        : closed (No error)
* -----------------------------------------------------------------------------
*/
```

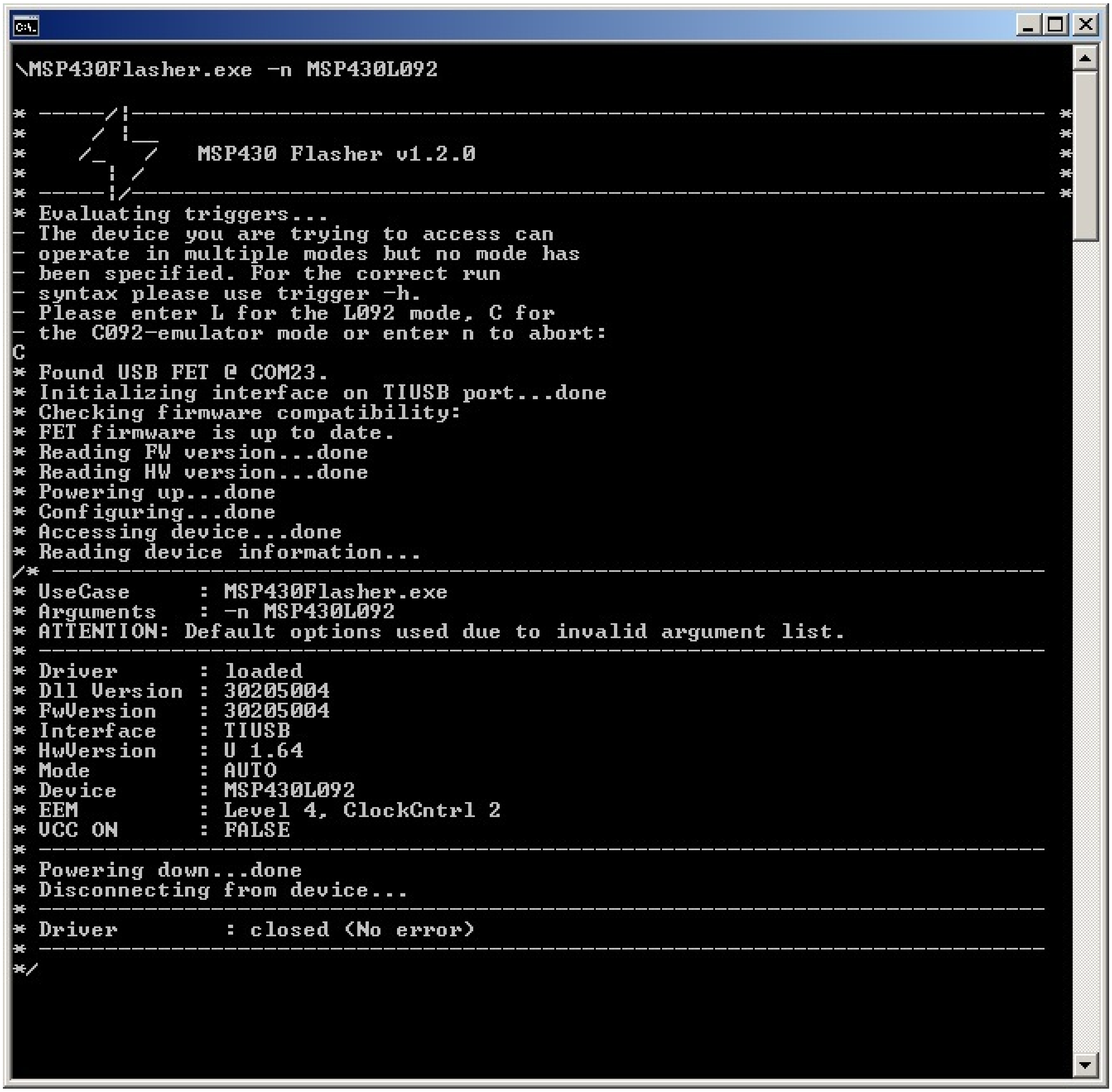
Figure 2. Reading Device Memory

7.3 Accessing a Device With a Device Activation Code

Some devices require a device activation code to be operable. Devices of this kind, such as the MSP430L092 or RF430 devices cause an error in MSP Flasher if the provided activation code is incorrect or if no activation code is provided. MSP Flasher provides the necessary Activation Code internally, but the user must specify the desired operating mode using the -o trigger. In the following example, this switch uses the argument L for the L092 operating mode (with external memory) and the argument C for the C092 operating mode (without external memory).

Figure 3 shows the console output after running the following command line:

```
MSP430Flasher -n MSP430L092
```



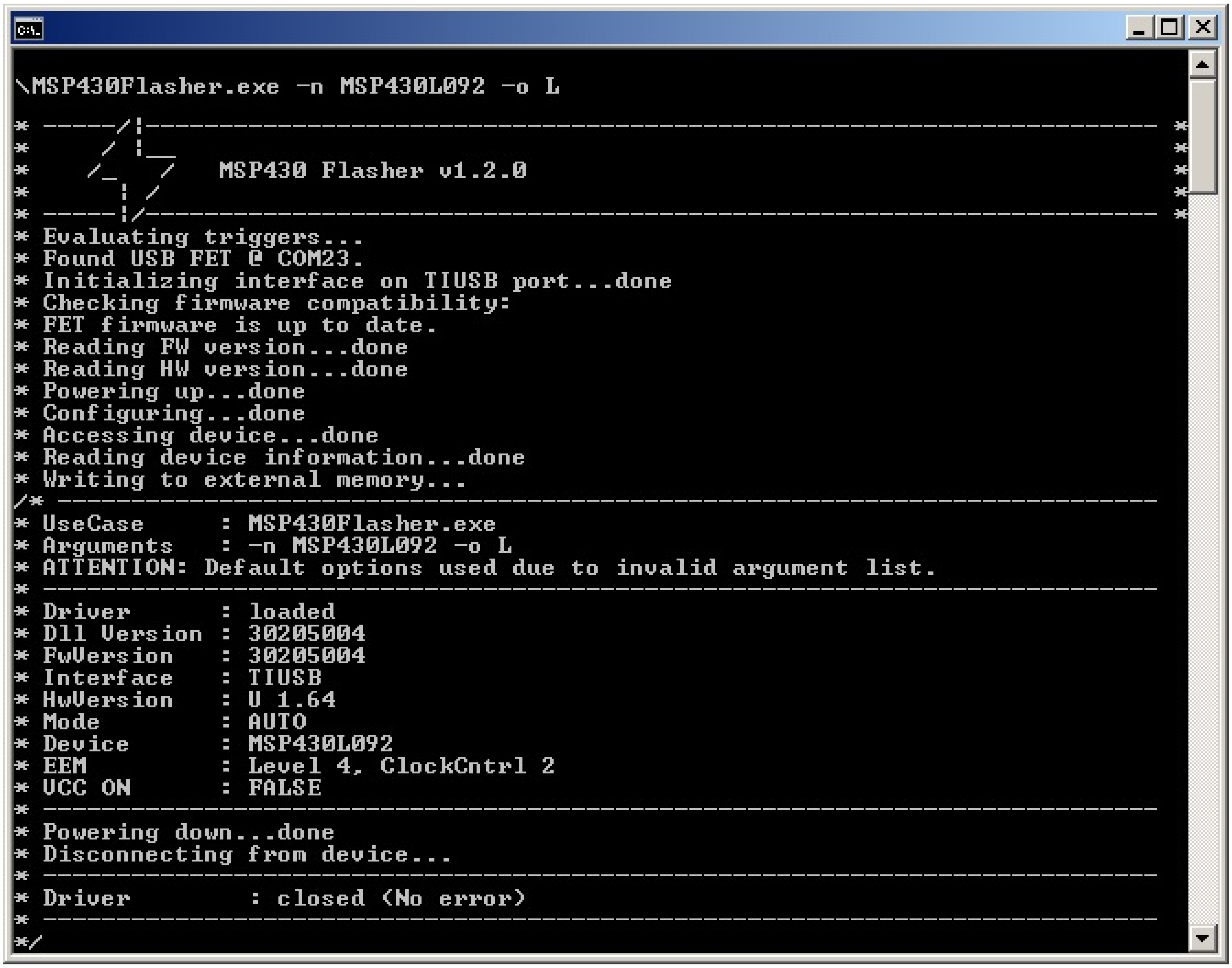
```
\MSP430Flasher.exe -n MSP430L092

* -----/|------------------------------------------------------------- *
*     / |__                                                           *
*    /_   /   MSP430 Flasher v1.2.0                                   *
*      | /                                                            *
* -----|/------------------------------------------------------------- *
* Evaluating triggers...
- The device you are trying to access can
- operate in multiple modes but no mode has
- been specified. For the correct run
- syntax please use trigger -h.
- Please enter L for the L092 mode, C for
- the C092-emulator mode or enter n to abort:
C
* Found USB FET @ COM23.
* Initializing interface on TIUSB port...done
* Checking firmware compatibility:
* FET firmware is up to date.
* Reading FW version...done
* Reading HW version...done
* Powering up...done
* Configuring...done
* Accessing device...done
* Reading device information...
/* ------------------------------------------------------------------
* UseCase      : MSP430Flasher.exe
* Arguments    : -n MSP430L092
* ATTENTION: Default options used due to invalid argument list.
* -------------------------------------------------------------------
* Driver       : loaded
* Dll Version  : 30205004
* FwVersion    : 30205004
* Interface    : TIUSB
* HwVersion    : U 1.64
* Mode         : AUTO
* Device       : MSP430L092
* EEM          : Level 4, ClockCntrl 2
* VCC ON       : FALSE
* -------------------------------------------------------------------
* Powering down...done
* Disconnecting from device...
* -------------------------------------------------------------------
* Driver         : closed (No error)
* -------------------------------------------------------------------
*/
```

Figure 3. Accessing an L092 Device Without Specifying an Operating Mode

MSP Flasher prompts to select the operating mode when the device name is found to be MSP430L092 and no mode has been selected. When C is entered as the device operating mode, the external memory is not accessed.

Figure 4 shows the console output after running the same command line with an additional -o switch to specify the operating mode.

`MSP430Flasher -n MSP430L092 -o L`



```
\MSP430Flasher.exe -n MSP430L092 -o L

* -----/|--------------------------------------------------------------------- *
*     / |__                                                                    *
*    /_   /   MSP430 Flasher v1.2.0                                            *
*      | /                                                                     *
* -----|/--------------------------------------------------------------------- *
* Evaluating triggers...
* Found USB FET @ COM23.
* Initializing interface on TIUSB port...done
* Checking firmware compatibility:
* FET firmware is up to date.
* Reading FW version...done
* Reading HW version...done
* Powering up...done
* Configuring...done
* Accessing device...done
* Reading device information...done
* Writing to external memory...
/* ---------------------------------------------------------------------------
* UseCase      : MSP430Flasher.exe
* Arguments    : -n MSP430L092 -o L
* ATTENTION: Default options used due to invalid argument list.
* ----------------------------------------------------------------------------
* Driver       : loaded
* Dll Version : 30205004
* FwVersion    : 30205004
* Interface    : TIUSB
* HwVersion    : V 1.64
* Mode         : AUTO
* Device       : MSP430L092
* EEM          : Level 4, ClockCntrl 2
* VCC ON       : FALSE
* ----------------------------------------------------------------------------
* Powering down...done
* Disconnecting from device...
* ----------------------------------------------------------------------------
* Driver         : closed (No error)
* ----------------------------------------------------------------------------
*/
```

Figure 4. Accessing a L092 Device

The L092 mode was selected from the start, so the user was not prompted for additional input. Note also that the MSP Flasher wrote to the external memory: "*Writing to external memory…*"

NOTE: If the -n switch is omitted, MSP Flasher cannot automatically detect whether an activation code is required and does not prompt the user to enter it.

7.4 Securing the Target Device

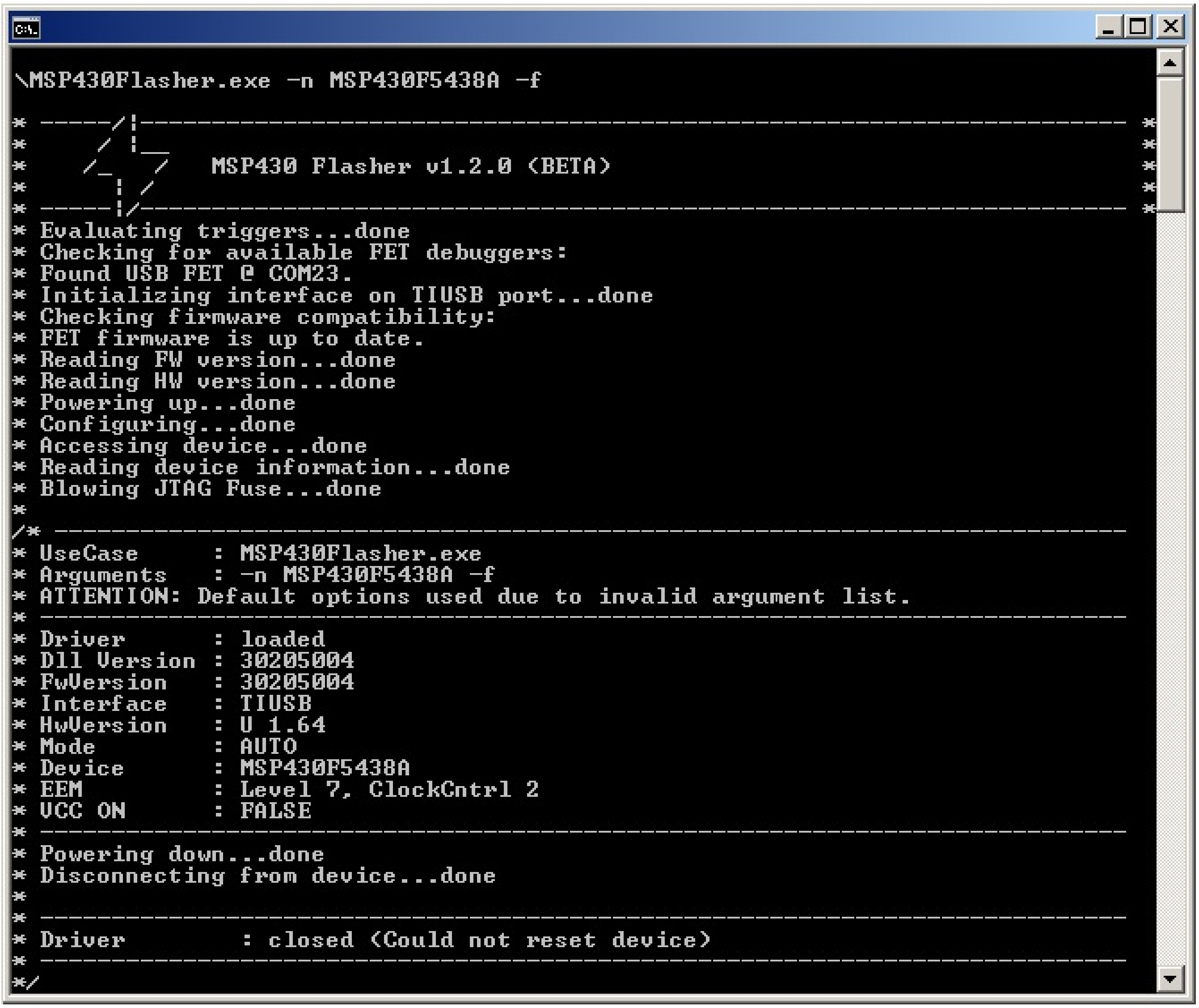
Use the -f switch to permanently lock JTAG access to the target device. For older MSPs from the 1xx, 2xx, and 4xx families, this trigger blows the internal poly fuse of the device, thus making the JTAG interface physically and irreversibly unusable. For newer MSPs (for example, from the 5xx and 6xx families) the -f switch programs the *electronic fuse* or *soft fuse* (see the device family user's guide for more details). For SimpleLink MSP432 devices, the security feature JTAG/SWD lock will be activated. If you re-connect to the device, a factory reset will be offered. A factory reset will erase main memory and reset all security settings on the device. To force a factory reset without prompt, use `-e ERASE_TOTAL`.

NOTE: Breakpoint functionality is disabled when the -f switch is used.

MSP Flasher cannot blow the JTAG security fuse of MSP430L092 devices.

```
MSP430Flasher -n MSP430F5438A -f
```

Figure 5 shows the console output after running the previous command line.

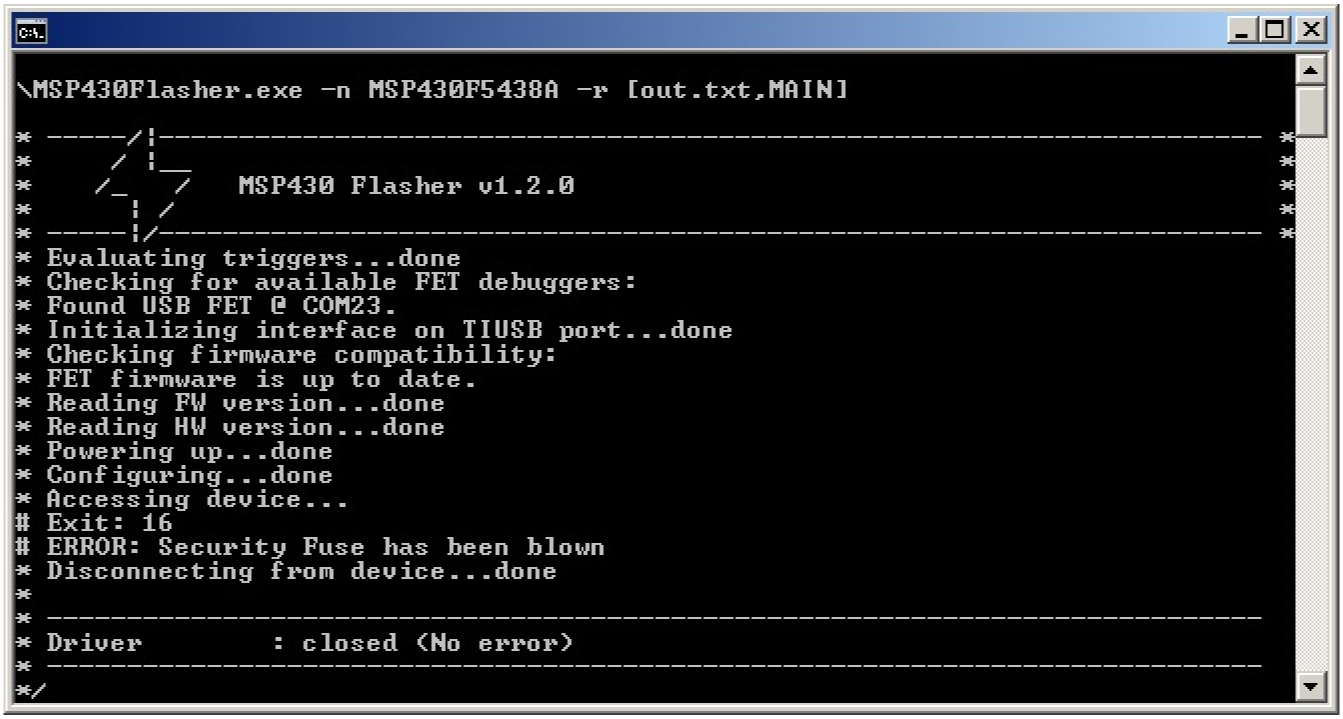


```
\MSP430Flasher.exe -n MSP430F5438A -f

* -----/|--------------------------------------------------------------------- *
*     / |__                                                                    *
*    /_   /  MSP430 Flasher v1.2.0 (BETA)                                      *
*      | /                                                                     *
* -----|/--------------------------------------------------------------------- *
* Evaluating triggers...done
* Checking for available FET debuggers:
* Found USB FET @ COM23.
* Initializing interface on TIUSB port...done
* Checking firmware compatibility:
* FET firmware is up to date.
* Reading FW version...done
* Reading HW version...done
* Powering up...done
* Configuring...done
* Accessing device...done
* Reading device information...done
* Blowing JTAG Fuse...done
*
/* ---------------------------------------------------------------------------
* UseCase      : MSP430Flasher.exe
* Arguments    : -n MSP430F5438A -f
* ATTENTION: Default options used due to invalid argument list.
* ----------------------------------------------------------------------------
* Driver       : loaded
* Dll Version  : 30205004
* FwVersion    : 30205004
* Interface    : TIUSB
* HwVersion    : U 1.64
* Mode         : AUTO
* Device       : MSP430F5438A
* EEM          : Level 7, ClockCntrl 2
* VCC ON       : FALSE
* ----------------------------------------------------------------------------
* Powering down...done
* Disconnecting from device...done
*
* ----------------------------------------------------------------------------
* Driver         : closed (Could not reset device)
* ----------------------------------------------------------------------------
*/
```

Figure 5. Securing the Target device

Figure 6 shows the console output after running the following command line to read the device main memory after securing the target device.

```
MSP430Flasher -n MSP430F5438A -r [out.txt,MAIN]
```



```
\MSP430Flasher.exe -n MSP430F5438A -r [out.txt,MAIN]

* -----/|------------------------------------------------------------------ *
*     / |__                                                                 *
*    /_   /   MSP430 Flasher v1.2.0                                         *
*      | /                                                                  *
* -----|/------------------------------------------------------------------ *
* Evaluating triggers...done
* Checking for available FET debuggers:
* Found USB FET @ COM23.
* Initializing interface on TIUSB port...done
* Checking firmware compatibility:
* FET firmware is up to date.
* Reading FW version...done
* Reading HW version...done
* Powering up...done
* Configuring...done
* Accessing device...
# Exit: 16
# ERROR: Security Fuse has been blown
* Disconnecting from device...done
*
* --------------------------------------------------------------------------
* Driver        : closed (No error)
* --------------------------------------------------------------------------
*/
```

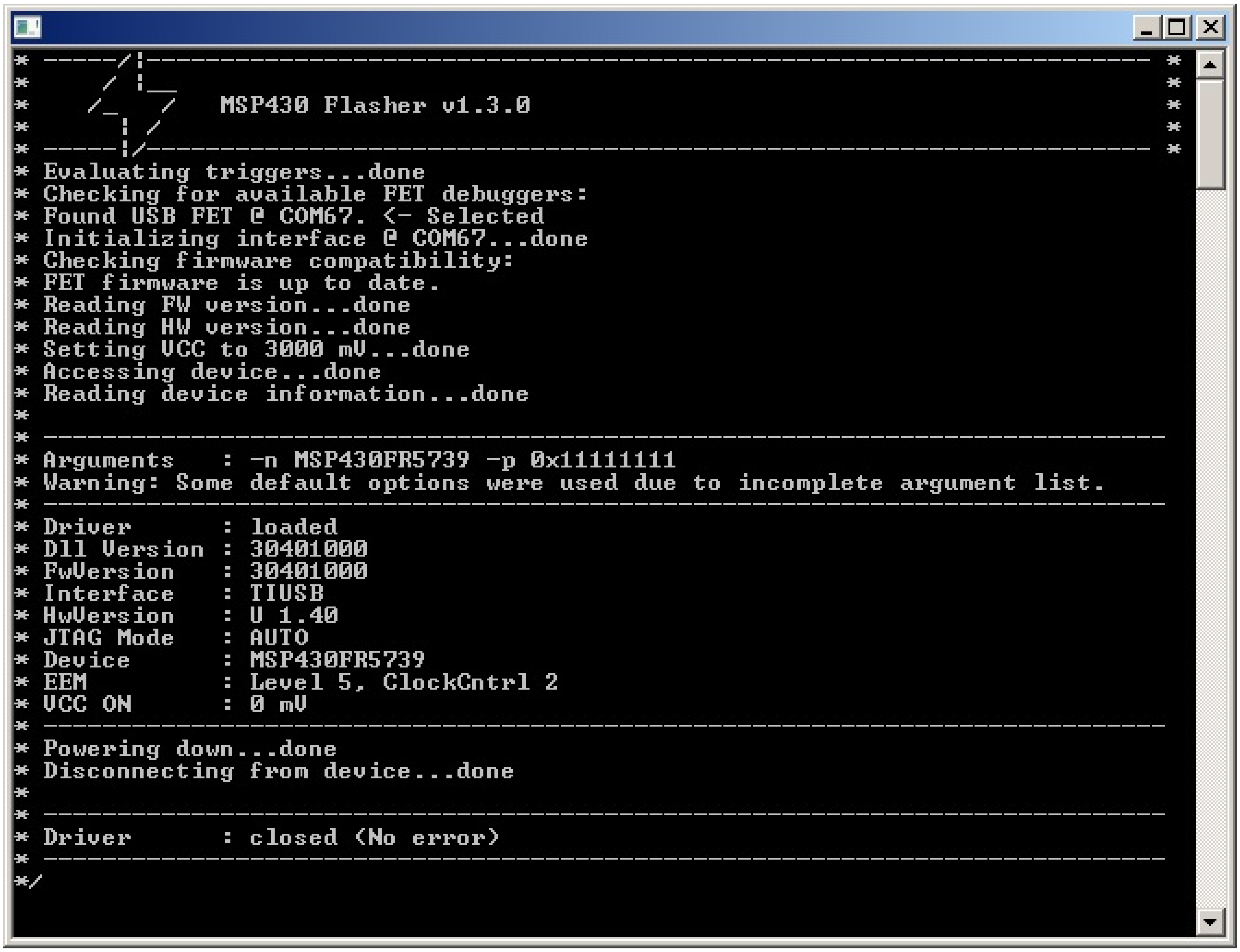
Figure 6. Trying to Access a Secured Target Device

7.5 Unlocking a Password-Protected Target Device

Newer MSP devicesfrom the FRxx families support a JTAG password lock mechanism that can be reversed by specifying a password (see the *MSP430FR57xx family user's guide*). This mechanism is not to be confused with the electronic fuse that permanently secures the JTAG interface.

To unlock a password-protected device, use the -p switch to provide the correct JTAG password (in hex format with a leading "0x"):

```
MSP430Flasher -n MSP430FR5739 -p 0x11111111
```



```
* -----/|-------------------------------------------------------------- *
*     / |__                                                             *
*    /_   /   MSP430 Flasher v1.3.0                                     *
*      | /                                                              *
* -----|/-------------------------------------------------------------- *
* Evaluating triggers...done
* Checking for available FET debuggers:
* Found USB FET @ COM67. <- Selected
* Initializing interface @ COM67...done
* Checking firmware compatibility:
* FET firmware is up to date.
* Reading FW version...done
* Reading HW version...done
* Setting VCC to 3000 mV...done
* Accessing device...done
* Reading device information...done
*
* ------------------------------------------------------------------------
* Arguments   : -n MSP430FR5739 -p 0x11111111
* Warning: Some default options were used due to incomplete argument list.
* ------------------------------------------------------------------------
* Driver      : loaded
* Dll Version : 30401000
* FwVersion   : 30401000
* Interface   : TIUSB
* HwVersion   : U 1.40
* JTAG Mode   : AUTO
* Device      : MSP430FR5739
* EEM         : Level 5, ClockCntrl 2
* VCC ON      : 0 mV
* ------------------------------------------------------------------------
* Powering down...done
* Disconnecting from device...done
*
* ------------------------------------------------------------------------
* Driver      : closed (No error)
* ------------------------------------------------------------------------
*/
```

Figure 7. Unlocking a Password-Protected Target Device

8 Using MSP Flasher on Unix

If multiple versions of libmsp430 are on the system, TI recommends invoking MSP Flasher by a script that sets the LD_LIBRARY_PATH. This method ensures that the libmsp430 library in the MSP Flasher installation directory is used.

Example:

```
#!/bin/bash
export LD_LIBRARY_PATH=.:$LD_LIBRARY_PATH
clear

./MSP430Flasher -w "Firmware.txt" -v -g -z [VCC]
read -p "Press any key to continue..."
./MSP430Flasher -r [FirmwareOutput.txt,MAIN]
read -p "Press any key to continue..."
```

9 Error Codes

Table 3 lists the possible error codes and messages.

Table 3. Error Codes

Error Code	Error Message
0	No error
1	Could not initialize device interface
2	Could not close device interface
3	Invalid parameter(s)
4	Could not find device (or device not supported)
5	Unknown device
6	Could not read device memory
7	Could not write device memory
8	Could not read device configuration fuses
9	Incorrectly configured device; device derivative not supported
10	Could not set device Vcc
11	Could not reset device
12	Could not preserve/restore device memory
13	Could not set device operating frequency
14	Could not erase device memory
15	Could not set device breakpoint
16	Could not single step device
17	Could not run device (to breakpoint)
18	Could not determine device state
19	Could not open Enhanced Emulation Module
20	Could not read Enhanced Emulation Module register
21	Could not write Enhanced Emulation Module register
22	Could not close Enhanced Emulation Module
23	File open error
24	File type could not be identified
25	File end error
26	File input/output error
27	File data error
28	Verification error
29	Could not secure the device
30	The Debug Interface to the device has been secured

Table 3. Error Codes (continued)

Error Code	Error Message
31	Error within Intel Hex file
32	Could not write device Register
33	Could not read device Register
34	Not supported by selected Interface or Interface is not initialized
35	Interface Communication error
36	No external power supply detected
37	External power too low
38	External power detected
39	External power too high
40	Hardware Self Test Error
41	Fast Flash Routine experienced a timeout
42	Could not create thread for polling
43	Could not initialize Enhanced Emulation Module
44	Insufficient resources
45	No clock control emulation on connected device
46	No state storage buffer implemented on connected device
47	Could not read trace buffer
48	Enable the variable watch function
49	No trigger sequencer implemented on connected device
50	Could not read sequencer state - Sequencer is disabled
51	Could not remove trigger - Used in sequencer
52	Could not set combination - Trigger is used in sequencer
53	System Protection Module A is enabled - Device locked
54	Invalid SPMA key was passed to the target device - Device locked
55	Device does not accept any further SPMA keys - Device locked
56	MSP-FET430UIF Firmware erased - Bootloader active
57	Could not find MSP-FET430UIF on specified COM port
58	MSP-FET430UIF is already in use
59	EEM polling thread is already active
60	Could not terminate EEM polling thread
61	Could not unlock BSL memory segments
62	Could not perform access, BSL memory segments are protected
63	Another device as selected was found
64	Could not enable JTAG wrong password
65	Only one UIF must be connected during update to v3
66	CDC-USB-FET-Driver was not installed. Please install the driver
67	Manual reboot of USB-FET needed ! PLEASE unplug and reconnect your USB-FET!!
68	Internal error
69	One of the connected MSP-FETs / eZ-FETs debuggers needs recovery
70	One of the connected MSP-FETs / eZ-FETs debuggers needs recovery
71	Feature not supported
72	Only one MSP-FET / eZ-FET must be connected during recovery
73	MSP-FET / eZ-FET recovery failed
74	MSP-FET / eZ-FET core(communication layer) update failed
75	MSP-FET / eZ-FET legacy module update failed
76	EnergyTrace is not supported by the selected debugger
77	Hardware State is unknown

Table 3. Error Codes (continued)

Error Code	Error Message
78	Device configuration data inconsistent. Please discontinue using/replace target device.
79	EEM module not accessible while running in Ultra Low Power Debug Mode - Deactivate Ultra Low Power Debug mode to enable this feature
80	Failed to remove software breakpoints, please reprogram target device
81	Trigger configuration conflicts with existing triggers
82	Operation not possible while device is running
83	This function can not be used when software breakpoints are enabled
84	JTAG/SBW speed configuration failed
85	Software breakpoint can't be set (followed by critical value)
86	EnergyTrace is not supported by selected MSP430 device
87	EnergyTrace requires Ultra-Low Power debug / LPMx.5 enabled
88	Legacy version of silicon used, which is no longer supported. Please contact TI to obtain a newer version.
89	Secure device via the IDE is not supported. See Device User Guide for further information.
90	Cycle counter is in basic mode. Set to advanced mode to use this function.
91	Parallel port FET (MSP-FETP430IF) is no longer supported.
92	Wrong target architecture was selected - Valid architectures are MSP430 or MSP432_M4.
93	Mass erase executed. Please power-cycle your device and restart the debug session.
94	Your connected hardware might drain too much power from the debugger.This results in an overcurrent.
95	MSP Tool firmware update failed. Please ensure the USB or Backchannel UART connection is not in use.
96	MSP432 devices are not supported using the MSPFET430-UIF
97	DAP is locked or wrong debug protocol selected.
98	Device database not loaded.
99	Invalid error number

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from November 16, 2017 to February 5, 2019	Page
• Added Section 9, *Error Codes*	14

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265