

# ELMB128 Lockbits & Fuses

## CONTENTS

1	RECOMMENDED LOCK BITS AND FUSE BITS SETTINGS.....	1
2	READING AND WRITING LOCK AND FUSE BITS VIA CAN.....	4

## 1 Recommended Lock Bits and Fuse Bits settings

The lock bits and fuse bits listed in the following tables show the default setting of the ATmega128 processor, as well as the recommended (and sometimes required) settings for the ELMB128 module when the ELMB128 is a single-processor module with an ATmega128 working in '128' mode (*not* '103' mode) and containing a Bootloader (for self-programming).

ATmega128 LOCK BITS				
Lock Bit	Bit No.	Description	Default	ELMB128
—	7	—	1 (unprogr)	1 (unprogr)
—	6	—	1 (unprogr)	1 (unprogr)
BLB12	5	<b>Boot lock bit</b> (no read access to Bootloader)	1 (unprogr)	1 (unprogr)
BLB11	4	<b>Boot lock bit</b> (no write access to Bootloader)	1 (unprogr)	0 (progr)
BLB02	3	<b>Boot lock bit</b> (no read access to Application)	1 (unprogr)	1 (unprogr)
BLB01	2	<b>Boot lock bit</b> (no write access to Application)	1 (unprogr)	1 (unprogr)
LB2	1	<b>Memory lock bit</b> (no read access to Flash or EEPROM)	1 (unprogr)	1 (unprogr)
LB1	0	<b>Memory lock bit</b> (no write access to Flash or EEPROM)	1 (unprogr)	1 (unprogr)
Byte value:			0xFF	0xEF

**Table 1. ATmega128 Lock Bits.** To write the lock bits to the ELMB128 as shown, send instruction 0xAC, 0xFF, 0x00, 0xEF (this is compatible with writing to ATmega103 leaving its 2 memory lockbits *unprogrammed*) or 0xAC, 0xE0, 0x00, 0xEF.

Read instruction for both 103/128: 0x58, 0x00, 0x00, 0x00; the lock bits are returned in the last byte of the reply (byte 7). **Default:** no memory protection.

**ELMB128:** Bootloader protected against writing to (BLB11 fuse)

<b>ATmega128 FUSE BITS (low)</b>				
<b>Lock Bit</b>	<b>Bit No.</b>	<b>Description</b>	<b>Default</b>	<b>ELMB128</b>
BODLEVEL	7	<b>Brown-out detector trigger level</b> (1=2.7V, 0=4.0V)	1 (unprogr)	1 (unprogr)
BODEN	6	<b>Brown-out detector enable</b>	1 (unprogr)	0 (progr)
SUT1	5	<b>Select start-up time</b>	1 (unprogr)	0 (progr)
SUT0	4	<b>Select start-up time</b>	0 (progr)	1 (unprogr)
CKSEL3	3	<b>Select clock source</b>	0 (progr)	0 (progr)
CKSEL2	2	<b>Select clock source</b>	0 (progr)	0 (progr)
CKSEL1	1	<b>Select clock source</b>	0 (progr)	0 (progr)
CKSEL0	0	<b>Select clock source</b>	1 (unprogr)	0 (progr)
Byte value:			0xE1	0x90

**Table 2. ATmega128 Fuse Bits (low byte).**

**Default:** no brown-out detection, 4 ms startup time (SUT0-1 fuses), internal 1MHz clock (CKSEL0-3 fuses).

**ELMB128:** brown-out detection enabled, maximum (64 ms) startup time (SUT0-1 fuses), external clock source (CKSEL0-3 fuses).

To write the fuse bits to the ELMB128 as shown, send instruction 0xAC, 0xA0, 0x00, 0xA0. Read instruction: 0x50, 0x00, 0x00, 0x00; the fuse bits are returned in the last byte of the reply (byte 7).

<b>ATmega103 FUSE BITS</b>				
<b>Lock Bit</b>	<b>Bit No.</b>	<b>Description</b>	<b>Default</b>	<b>ELMB</b>
—	7	—	1	1
—	6	—	0	0
—	5	—	1	1
—	4	—	1	1
EESAVE	3	<b>Preserve EEPROM on Chip Erase</b>	1 (unprogr)	0 (progr)
—	2	—	1	1
SUT1	1	<b>Select start-up time</b>	1 (unprogr)	1 (unprogr)
SUT0	0	<b>Select start-up time</b>	1 (unprogr)	1 (unprogr)
Byte value:			0xBF	0xB7

**Table 3. ATmega103 Fuse Bits.**

Startup time about 16 ms (SUT0-1 fuses as shown).

To write the fuse bits to the ELMB as shown, send instruction 0xAC, 0xB7, 0x00, 0x00.

Read instruction is compatible with ATmega128: 0x50, 0x00, 0x00, 0x00; the fuse bits are returned in the last byte of the reply (byte 7).

<b>ATmega128 FUSE BITS (high)</b>				
<b>Lock Bit</b>	<b>Bit No.</b>	<b>Description</b>	<b>Default</b>	<b>ELMB128</b>
OCDEN	7	Enable OCD	1 (unprogr)	1 (unprogr)
JTAGEN	6	Enable JTAG	1 (unprogr)	1 (unprogr)
SPIEN	5	Enable serial program and data downloading	0 (progr)	0 (progr)
CKOPT	4	Oscillator options	1 (unprogr)	0 (progr)
EESAVE	3	Preserve EEPROM on Chip Erase	1 (unprogr)	0 (progr)
BOOTSZ1	2	Select Bootloader size	0 (progr)	0 (progr)
BOOTSZ0	1	Select Bootloader size	0 (progr)	0 (progr)
BOOTRST	0	Select Reset Vector	1 (unprogr)	0 (progr)
Byte value:			0xD9	0xC0

**Table 4. ATmega128 Fuse Bits (high byte).**

**Default:** Bootloader size 4 Kwords (8 Kbyte), Application Reset.

**ELMB128:** Bootloader size 4 Kwords (8 Kbyte), Bootloader Reset, internal 36 pF capacitor between XTAL1 and GND enabled (CKOPT fuse).

To write the fuse bits to the ELMB128 as shown, send instruction 0xAC, 0xA8, 0x00, 0xD0. Read instruction: 0x58, 0x08, 0x00, 0x00; the fuse bits are returned in the last byte of the reply (byte 7).

<b>ATmega128 FUSE BITS (extended)</b>				
<b>Lock Bit</b>	<b>Bit No.</b>	<b>Description</b>	<b>Default</b>	<b>ELMB128</b>
—	7	—	1	1
—	6	—	1	1
—	5	—	1	1
—	4	—	1	1
—	3	—	1	1
—	2	—	1	1
M103C	1	ATmega103 compatibility mode	0 (progr)	1 (unprogr)
WDTON	0	Watchdog Timer always on	1 (unprogr)	0 (progr)
Byte value:			0xFD	0xFE

**Table 5. ATmega128 Fuse Bits (extended).**

**Default:** ATmega128 in ATmega103 compatibility mode, Watchdog Timer under software control

**ELMB128:** native ATmega128 mode, Watchdog Timer always on.

To write the fuse bits to the ELMB128 as shown, send instruction 0xAC, 0xA4, 0x00, 0xFE. Read instruction: 0x50, 0x08, 0x00, 0x00; the fuse bits are returned in the last byte of the reply (byte 7).

## 2 Reading and Writing Lock and Fuse Bits via CAN

NB: writing fuse bits via CAN/ATmega128-Bootloader is not possible !

Normally the ATmega128 processor of the ELMB128 module has been preconfigured (lock and fuse bits) and preloaded with at least a Bootloader, but a user might want to change particular settings to suit his requirements.

Table 6 shows the 4-byte instructions which have to be written to Object 1F50, subindex 1 of the MDT-DCS *Object Dictionary* by *SDO* message, to read or write ATmega128 (or ATmega103) processor lock bits or fuse bits (NB: values written represent the recommended settings presented in the previous section). The byte containing the bits being set are underlined (note that ATmega128 has these bits in the 4<sup>th</sup> byte and ATmega103 in the 2<sup>nd</sup> byte of the message).

The bytes as shown from left to right must be put in data bytes 4 to 7 of the *SDO*-client CAN-message (the message sent to the Slave processor or Bootloader) In brackets the same bytes are shown as a single 32-bit integer number.

If a read operation is performed the bits are returned in the last byte of the *SDO*-server reply (byte 7), which is the message sent in reply by the Bootloader (or Slave processor).

In case the Slave processor is executing the programming instructions (instead of a Bootloader) it is necessary before sending any of the lockbits/fusebits instructions to send a *Programming Enable* instruction (0xAC, 0x53, 0x00, 0x00) . If the *SDO*-server reply contains in data byte 6 a copy of the value of byte 5, i.e. 0x53, the Programming Enable instruction was successfully completed., and lock bits and fuse bits read/write instructions can now be given.

Programming instructions can be composed and sent using any host (PC) program for generating CAN(open) messages (i.e. *SDOs*), such as *CANalyzer* (from the VECTOR company) or *OPENhost* (from NIKHEF).

	<b>ATmega128</b>		<b>ATmega103</b>	
	<b>Read</b>	<b>Write</b>	<b>Read</b>	<b>Write</b>
<b>Lock Bits</b>	58 00 00 00 (00000058)	AC FF 00 <u>EF</u> (EF00FFAC)	58 00 00 00 (00000058)	AC <u>FF</u> 00 EF (FF00FFAC)
<b>Fuse Bits (lo)</b>	50 00 00 00 (00000050)	AC A0 00 <u>A0</u> (A000A0AC)	50 00 00 00 (00000050)	AC <u>B7</u> 00 00 (0000B7AC)
<b>Fuse Bits (hi)</b>	58 08 00 00 (00000858)	AC A8 00 <u>D0</u> (D000A8AC)	—	—
<b>Fuse Bits (ext)</b>	50 08 00 00 (00000850)	AC A4 00 <u>FE</u> (FE00A4AC)	—	—

**Table 6.** List of 4-byte instructions for reading and writing ELMB(128) Lock Bits and Fuse Bits (numbers in hexadecimal). Underlined bytes are the actual lock or fuse bits being set (which may differ from shown values depending on the required configuration). Note that writing fuse bits is not possible via an ATmega128 Bootloader (the instructions with grey background).