Digilent AVR Programmer™ Reference Manual

1/31/2006

www.digilentinc.com
215 E Main Suite D | Pullman, WA 99163
(509) 334 6306 Voice and Fax

Introduction

The Digilent AVR Programmer is a Microsoft® Windows® application used for in-system programming of Atmel AVR® microcontrollers. It is specifically designed for programming AVR devices on Digilent products, but is also a general purpose AVR device programmer for those AVR devices that support the serial in-system programming protocol.

The AVR Programmer supports the in-system programming of Atmel AVR devices using a programming cable connected to a 6-pin programming connector on the target board. All Digilent products with a AVR microcontroller have a suitable connector.

Installation Information

The AVR Programmer is installed by running the Windows installer application available for download from the Digilent web site.

The installer will copy the application program files, required device drivers, AVR Part Description files, and documentation file to the user's computer.

The part description files are copied to the folder: Program Files\Common Files\Digilent\AvrParts.

Overview of AVR Microcontrollers

The Atmel AVR devices are 8-bit microcontrollers with various I/O peripheral functions and varying amounts of program and data memory.

There are two different ways that AVR devices can be programmed. High-voltage parallel programming involves placing the AVR device into a special device programmer. This can only be done before the device is soldered to a PC board. In-system programming uses a serial interface and an in-system programming protocol to program the devices in circuit. Any AVR device can be programmed using the parallel programming method. Most, but not all, AVR devices also support the in-system programming method. The AVR Programmer only works with in-system programming.

There are four basic parts of an AVR device that can be programmed: the program flash, the data EEPROM, fuses, and lock bits.

The program flash holds program instructions to be executed. The data EEPROM is non-volatile memory that can be programmed via the in-system programming protocol, and can also be read or written by the program running on the device. Program flash and data EEPROM are configured using the Program tab in the AVR Programmer.

Fuses and lock bits are used to select the device's operating modes. They are described in more detail below.

Before programming the flash or EEPROM memory, you must first erase the device. When flash or EEPROM memory are erased, the bits are set to the 1 state. When programming, 0 bits are programmed to 0 and 1 bits are left unchanged. It is not possible to program a bit from 0 to 1. This can only be accomplished by erasing.

Flash and EEPROM memory can only be erased and reprogrammed a limited number of times before they wear out. Atmel rates the memories in the AVR parts as having an endurance of 10,000 erase/reprogram cycles. Programming over already-programmed bits will wear them out prematurely. The AVR Programmer allows you to manually erase the device and has an option to automatically erase the device while programming.

The file format used to hold the data for programming the flash and EEPROM memories is the Intel HEX file format. Intel HEX is an industry standard file format used for device programming files. Software development tools designed to work with AVR parts produce Intel HEX format files. There are two files produced by the development tools. One contains the data to be programmed into the program flash memory, and the other contains the data to be programmed into the data EEPROM. When the AVR Programmer is used to read back the contents of device memory, an Intel HEX format file is generated.

Fuses are control bits that establish the basic operating modes of the device. These control such things as the clock source, enabling the watchdog timer, setting compatibility modes, and so on. The number and meaning of the fuse bits vary between the different AVR devices. Device fuses are configured using the Fuses tab in the AVR Programmer.

Warning! Use caution when setting the clock source in the fuses. The in-system programming protocol uses the clock source currently selected by the fuses to clock the internal programming state machine. If the fuses are set to select a non-functioning clock source (such as the external crystal oscillator when no crystal is provided on the board), the in-system programming interface will stop functioning. This will render the board/device unusable and usually requires unsoldering the device from the board and replacing it with a new, un-programmed, device. AVR devices come from the factory with the fuses set to use an internal oscillator to guarantee that the in-system programming interface will work in all circumstances.

The lock bits are used to lock different parts of the device memory. This is done to prevent reprogramming and/or reading back the contents of memory to prevent the altering or reverse engineering of a design. Once a lock bit is set, that portion of the memory can't be reprogrammed or read back, depending on the specific lock bit setting, until the entire device is erased. The number and meaning of the lock bits vary between different AVR devices. Lock bits are configured using the Lock Bits tab in the AVR Programmer.

Overview of the AVR Programmer

Communication

The AVR in-system programming protocol uses the serial peripheral interface (SPI) for the communications channel between the PC and the device being programmed. Digilent has two different programming cables that can be used to provide the communications interface: the Digilent Parallel JTAG cable and the Digilent JTAG-USB cable.

Digilent programming cables conform to the JTAG standard and therefore bear the JTAG signal names instead of the SPI signal names. Observe the following label conventions when connecting the cable to a board:

TMS = SS or RST
TDI = MOSI
TDO = MISO
TCK = SCK

The parallel cable connects to the parallel printer port on a PC and to the Digilent 6-pin programming connector on the board being programmed. The AVR Programmer implements the SPI interface over the parallel port. The GIVEIO.SYS device driver must be installed to provide access to the parallel port. GIVEIO.SYS is automatically installed when the AVR programmer is installed.

The JTAG-USB cable connects a USB port on the PC and the 6-pin connector on the board being programmed. Firmware in the USB cable provides the SPI implementation. The AVR Programmer communicates with the USB cable using the drivers provided for that cable (specify the cable on the Programmer Settings tab).

Device Parameters

The AVR Programmer is designed to work with any AVR device that supports the in-system programming protocol. In order to work with different devices, the programmer needs to know certain properties of the device being programmed. This information can be provided in one of three ways: Digilent part description files, Atmel part description files, or manual settings.

Digilent part description files contain the required device programming information, plus additional information used to describe the fuses and lock bits available. Digilent makes available a part description file for the AVR devices used on Digilent product boards. These are installed when the AVR Programmer is installed. Additional or updated device files can be downloaded from the Digilent web site. The *Digilent AVR Part Description File Specification* document is also available for download. It describes the syntax conventions used in a Digilent part description file so that you can create part description files. See www.digilentinc.com for more information.

Atmel part description files are installed when the Atmel AVR Studio application is installed. These files contain the required device information and some of the additional information that describes the fuses and lock bits available in the device.

If no part description file is available for a device, you can enter the required information using the AVR Device Manual Settings fields on the Device Settings tab. The information to be entered in these fields can be found in the Memory Programming section of the part data sheet for the specific AVR device. AVR part data sheets can be downloaded from Atmel at www.atmel.com. When manual settings are being used, the list boxes in the Fuses and Lock Bits tabs will be empty. The fuses and lock bits values must be entered as hexadecimal values in the Value field. Complete descriptions for the fuses and lock bits for a device are found in the Memory Programming section of the device data sheet.

The AVR Device drop-down list box on the Program tab lists the available device files and the manual settings option. Manual Settings is listed first, followed by the available Digilent part description files, and then the Atmel part description files.

The part description files are listed by file name in the drop-down list box. If two files having the same name appear in the list, the first one found will be used. Since the Digilent files are looked for before the Atmel files, a Digilent file will override an Atmel file having the same name.

Signature Bytes

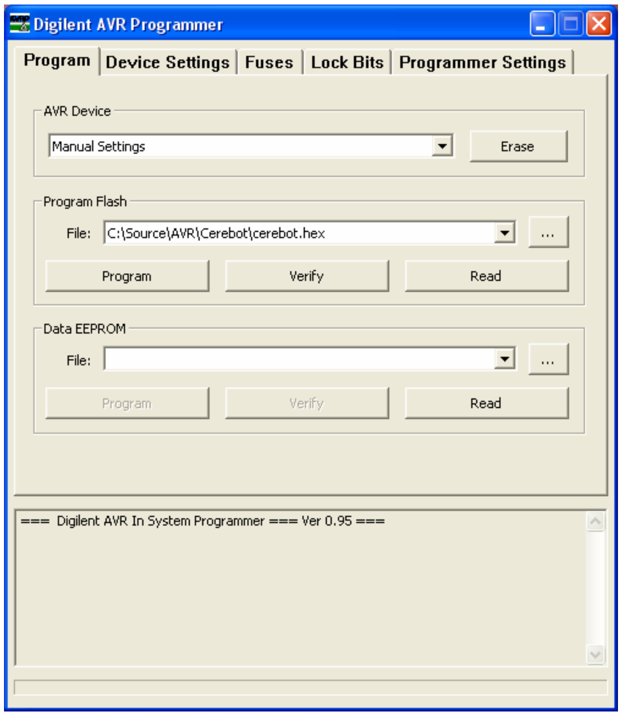
Each AVR device contains a device signature. This is a three-byte sequence that identifies the specific device. Before any programming operation, the AVR programmer checks the signature bytes read from the device with the signature bytes for the selected device. If the signatures don’t match no action is performed. This prevents accidentally damaging a device by programming it with data for a different part.

AVR Programmer User Interface

The user interface of the AVR Programmer has two main parts. A tabbed dialog box occupies the upper portion of the window with tabs for the major functional areas of the program. The lower portion of the window features a scrolling status box that gives feedback on the state of operation of the program.

Program Tab

The Program tab is used for device type selection and configuring the memories in the device. It is divided into three sections: AVR Device, Program Flash, and Data EEPROM.



AVR Device: Manual Settings

Program Flash

File: C:\Source\AVR\Cerebot\cerebot.hex

Data EEPROM

File:

```
=== Digilent AVR In System Programmer === Ver 0.95 ===
```

AVR Device Section

The **AVR Device** drop-down list box is used to select the part description file for the AVR device being programmed, or to choose manual device settings. Choose the appropriate selection from the available items in the list.

This list box contains the Manual Settings choice, followed by the available Digilent part description files and the available Atmel part description files.

The **Erase** button erases the memory in the device. For most AVR devices this erases both the program flash and the data EEPROM. Some devices have a fuse bit that disables EEPROM Erase when the device is erased. In this case erasure of the EEPROM depends on the setting of this fuse bit.

Program Flash Section

The **File** drop down list box contains up to eight of the most recently selected programming files. The file shown in this list box is used to program the device.

The **browse** button is marked with an ellipsis (…) and displays an Open dialog for browsing the file system to find a programming file.

The **Program** button programs the contents of the selected programming file into the program flash memory on the device. Depending on the options selected on the Programmer Settings tab, the device may be automatically erased before programming and the device contents may be read and verified after programming. The status window reports the actions taken and their results.

The **Verify** button reads the contents of the program flash memory and compares it against the contents of the selected programming file. This can be used to verify that the device was programmed correctly or to determine if the device is currently programmed with the contents of a specific file. The verify operation may not work correctly if the settings of the lock bits prevent reading back part of the device memory.

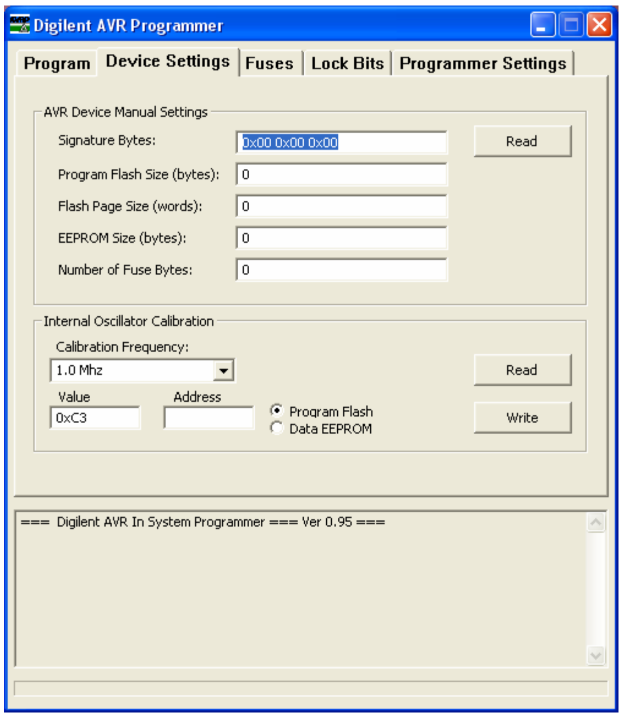
The **Read** button reads the contents of the program flash memory and writes it to a file. When the Read button is clicked, a Save As dialog box is displayed so that the file can be named. When the Save button on the Save File box is clicked, the contents of the program flash memory are read, converted to Intel HEX format and written to the specified file. This file can then be used to program another device. The read operation may not work correctly if the settings of the lock bits prevent reading back part of the device memory.

Data EEPROM Section

The Data EEPROM section contains the same user interface elements with the same functionality as the Program Flash section.

Device Settings Tab

The Device Settings tab is used to examine or set the device properties used to control the device programming process. The tab also contains the controls used to manage the Internal Oscillator Calibration feature.



AVR Device Manual Settings

Signature Bytes:	0x00 0x00 0x00
Program Flash Size (bytes):	0
Flash Page Size (words):	0
EEPROM Size (bytes):	0
Number of Fuse Bytes:	0

Internal Oscillator Calibration

Calibration Frequency: 1.0 Mhz

Value: 0xC3

Address:

[✓ Program Flash] Data EEPROM

```
=== Digilent AVR In System Programmer === Ver 0.95 ===
```

AVR Device Manual Settings Section

This section includes several fields containing the device properties that control the device programming process. If Manual Settings is chosen in the AVR Device list box on the Program tab, then these fields will be active and you must enter the appropriate values into these fields. If a part description file is chosen instead of Manual Settings, then these fields will be inactive and will display the information read from the part description file.

For entering the settings manually, the information needed for these fields can be found in the “Memory Programming” section of the data sheet for the specific AVR device.

The **Signature Bytes** field contains the three-byte signature for the device. If entered manually, it is entered as three hexadecimal values separated by spaces. For example, the signature for an ATmega64 is entered as *0x1E 0x96 0x02* .

The **Read** button causes the signature to be read from the device and displayed in the Signature Bytes field.

The **Program Flash Size (bytes)** field specifies the size of the program flash memory. It is entered as a decimal integer number.

The **Flash Page Size (words)** field specifies the size of the program flash page buffer. Some AVR devices use a page buffer to hold pages of program data during the programming process. This is entered as a decimal integer number giving the size of the page buffer in 16-bit words. For AVR devices that are not paged, the decimal value 1 should be placed in this field.

The **EEPROM Size (bytes)** field specifies the size of the data EEPROM memory. It is entered as a decimal integer number.

The **Number of Fuse Bytes** field specifies the number of fuse bytes in the device. AVR devices have between one and three fuse bytes, depending on the specific device.

Internal Oscillator Calibration Section

Most AVR devices have an internal RC oscillator available as one of the clock sources. This oscillator can be calibrated using calibration values written into each device at the factory. A calibration value is a byte value that can be written to the calibration register to tune the RC oscillator to operate more closely at its nominal frequency. The calibration values are stored in a special non-volatile memory that is not accessible at run time in all devices.

If the oscillator calibration feature is used, it is typical for the calibration value to be read from the device and stored in a known location in memory while programming the device. The calibration value will then be read from the known memory location and written to the calibration register during the startup initialization of the firmware program running on the device.

The Internal Oscillator Calibration user interface provides facilities for reading a calibration value from the device and writing it to a specified location in either the program flash memory or the data EEPROM memory.

The **Calibration Frequency** drop-down list box contains the list of calibration values available for the device.

The **Read** button reads the selected calibration value and displays it in the Value field.

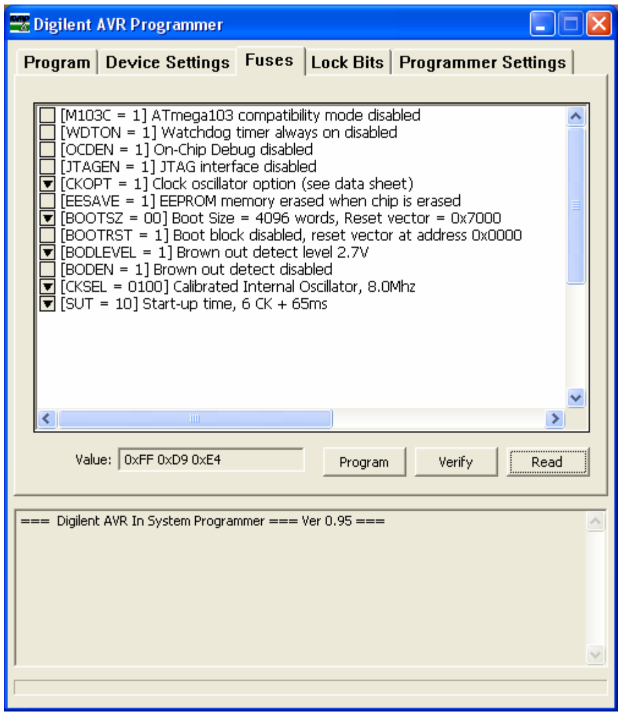
The **Value** field displays the calibration value read from the device or the value to be written to the device. If entered manually, it is entered as a single hexadecimal byte value.

The **Address** field specifies the memory address at which the calibration value will be written. The Program Flash and Data EEPROM radio buttons select the memory to which this address refers.

The **Write** button writes the byte displayed in the Value field to the specified memory address.

Fuses Tab

The Fuses tab is used to read and set the fuse bits in the device. This tab includes a fuses list, a Value field, and some control buttons.



☐ [M103C = 1] ATmega103 compatibility mode disabled
☐ [WDTON = 1] Watchdog timer always on disabled
☐ [OCDEN = 1] On-Chip Debug disabled
☐ [JTAGEN = 1] JTAG interface disabled
☑ [CKOPT = 1] Clock oscillator option (see data sheet)
☐ [EESAVE = 1] EEPROM memory erased when chip is erased
☑ [BOOTSZ = 00] Boot Size = 4096 words, Reset vector = 0x7000
☐ [BOOTRST = 1] Boot block disabled, reset vector at address 0x0000
☑ [BODLEVEL = 1] Brown out detect level 2.7V
☐ [BODEN = 1] Brown out detect disabled
☑ [CKSEL = 0100] Calibrated Internal Oscillator, 8.0Mhz
☑ [SUT = 10] Start-up time, 6 CK + 65ms

Value: 0xFF 0xD9 0xE4

```
=== Digilent AVR In System Programmer === Ver 0.95 ===
```

The **fuses list** displays the available sets of fuse bits as described by the selected part description file. The contents of the fuses list varies depending on the device selected in the AVR Device field of the Program tab. Refer to the device's Atmel data sheet for a description of its fuse bits. If Manual Settings is chosen for the AVR Device, then this list will be empty.

The **Value field** shows the binary value for the fuse bytes that corresponds to the current selections made in the fuses list. The number of bytes shown corresponds to the number of fuse bytes defined for the selected AVR device.

If a part description file is selected in the AVR Device field of the Program tab, this field is inactive and merely reports the binary value corresponding to the selections in the fuses list. If Manual Settings is chosen in the AVR Device field of the Program tab, this field is active and the fuses list will be empty. In this case, enter the desired value for the fuses as one to three hexadecimal values (depending on the number of fuse bytes in the device).

Note that for most fuse bits, setting the bit to 0 enables the function. The values are entered with the extended fuse byte first, followed by the high fuse byte and then the low fuse byte.

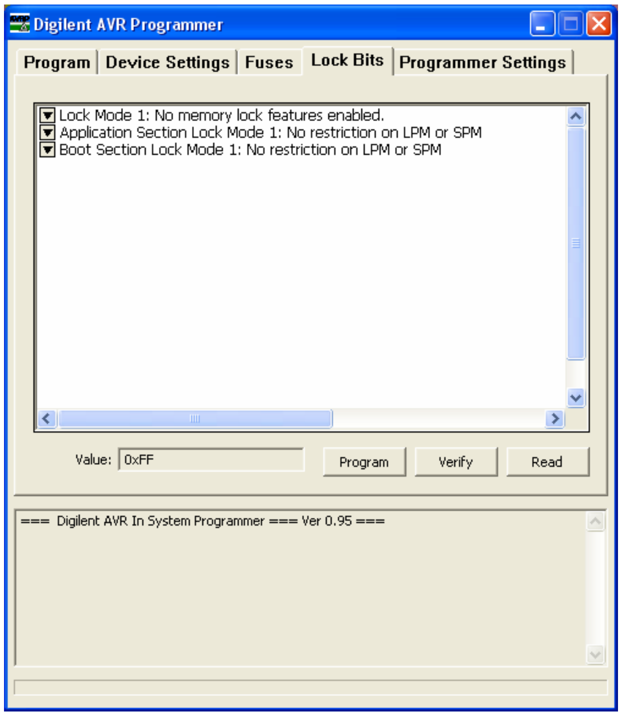
The **Program** button programs the currently selected fuses into the device.

The **Verify** button reads the current value of the fuses from the device and compares them against the selections in the fuses list.

The **Read** button reads the value of the fuse bits from the device and updates the fuses list and value field.

Lock Bits Tab

The Lock Bits tab is used to read and set the lock bits in the device. The contents and operation of the Lock Bits tab is similar to the Fuses tab.



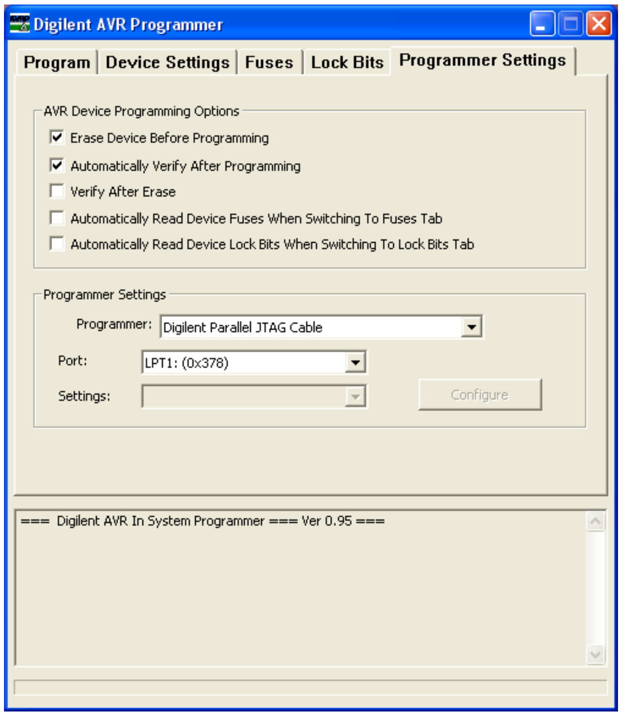
Lock Mode 1: No memory lock features enabled.
Application Section Lock Mode 1: No restriction on LPM or SPM
Boot Section Lock Mode 1: No restriction on LPM or SPM

Value: 0xFF

```
=== Digilent AVR In System Programmer === Ver 0.95 ===
```

Programmer Settings Tab

The Programmer Settings tab is used to select options to control the operation of the AVR Programmer. The AVR Device Programming Options are used to enable or disable different modes of operation of the Programmer. This tab is also used to select the communications interface to be used to talk to the device being programmed.



AVR Device Programming Options
- [✓] Erase Device Before Programming
- [✓] Automatically Verify After Programming
- [ ] Verify After Erase
- [ ] Automatically Read Device Fuses When Switching To Fuses Tab
- [ ] Automatically Read Device Lock Bits When Switching To Lock Bits Tab

Programmer Settings
Programmer: Digilent Parallel JTAG Cable
Port: LPT1: (0x378)
Settings:

```
=== Digilent AVR In System Programmer === Ver 0.95 ===
```

AVR Device Programming Options Section

The **Erase Device Before Programming** checkbox causes the device to be erased before the flash memory is programmed. No erase is performed automatically before programming the data EEPROM, so if this option is used, you should program the flash memory before programming the EEPROM. If this option is disabled, the device should be erased manually before programming.

The **Automatically Verify After Programming** checkbox causes the device memory (either flash or EEPROM) to be read and compared against the contents of the programming file after programming. Any errors are reported in the status window.

The **Verify After Erase** checkbox causes the contents of the program flash and data EEPROM to be read back after an erase operation to verify that the device is properly erased.

The **Automatically Read Device Fuses When Switching To Fuses Tab** checkbox causes the current fuse setting in the device to be read and used to set the options in the fuse list each time the fuses tab is displayed.

The **Automatically Read Device Lock Bits When Switching To Lock Bits Tab** checkbox causes the current lock bits setting in the device to be read and used to set the options in the lock bits list each time the lock bits tab is displayed.

Programmer Settings Section

The ***Programmer*** drop-down list box specifies the Digilent programming cable to use for communication with the device being programmed.

The **Port** drop-down list box specifies the port/device to use from the available devices installed in the system. This list box is populated with the corresponding choices for the cable selected in the Programmer field.

The **Settings** drop-down list box specifies the additional settings needed to configure the selected communications device.

The **Configure** button, if enabled, displays a dialog box for configuring the programming cable.