Chipsmall
深圳市津利帝科技有限公司
Geelead Technology Limited
Chipsmall
采购部

AVR® STK500

User Guide

Table of Contents

Section 1

Introduction 1-1

1.1 Starter Kit Features 1-1

1.2 Device Support 1-2

Section 2

Getting Started 2-1

2.1 Unpacking the System 2-1

2.2 System Requirements...2-1

2.3 Quick Start 2-2

2.3.1 Connecting the Hardware 2-3

2.3.2 Programming the Target AVR Device 2-3

Section 3

Hardware Description 3-1

3.1 Description of User LEDs..3-1

3.2 Description of User Switches 3-2

3.3 Connection of LEDs and Switches...3-3

3.4 Port Connectors 3-3

3.5 Description of User RS-232 Interface 3-5

3.6 Description of DataFlash Pins..3-6

3.7 Target Socket Section...3-8

3.7.1 ISP Programming 3-9

3.7.2 High-voltage Programming...3-11

3.8 Jumper Settings 3-15

3.8.1 Target V_{CC} Settings, VTARGET 3-16

3.8.2 Analog Reference Voltage, AREF 3-17

3.8.3 Reset Settings, RESET 3-18

3.8.4 Clock Settings, XTAL1 and OSCSEL 3-20

3.8.5 BSEL2 Jumper 3-22

3.8.6 PJUMP Jumpers...3-22

3.9 Expansion Connectors..3-23

3.9.1 Signal Descriptions...3-25

3.10 Prog Ctrl and Prog Data Headers 3-25

3.11 Miscellaneous 3-26

3.11.1 RESET Push Button 3-26

3.11.2 PROGRAM Push Button 3-27

3.11.3 Main Power LED....3-27
3.11.4 Target Power LED....3-27
3.11.5 Status LED....3-27

Section 4

Installing AVR Studio....4-1

Section 5

Using AVR Studio....5-1
5.1 Windows Software....5-1
5.2 Starting the Windows Software....5-1
5.2.1 Starting STK500....5-1
5.3 STK500 User Interface....5-2
5.3.1 “Program” Settings....5-2
5.3.2 “Fuses” Settings....5-3
5.3.3 “LockBits” Settings....5-4
5.3.4 “Advanced” Settings....5-4
5.3.5 “Board” Settings....5-5
5.3.6 “Auto” Settings....5-7
5.3.7 History Window....5-8
5.4 Command Line Software....5-9
5.5 Parameters....5-9

Section 6

In-System Programming of an External Target System....6-1

Section 7

Troubleshooting Guide....7-1

Section 8

Technical Support....8-1

Section 9

Example Applications....9-1
9.1 Using LEDs and Switches....9-1

Section 10

Appendix A....10-1

Section 1

Introduction

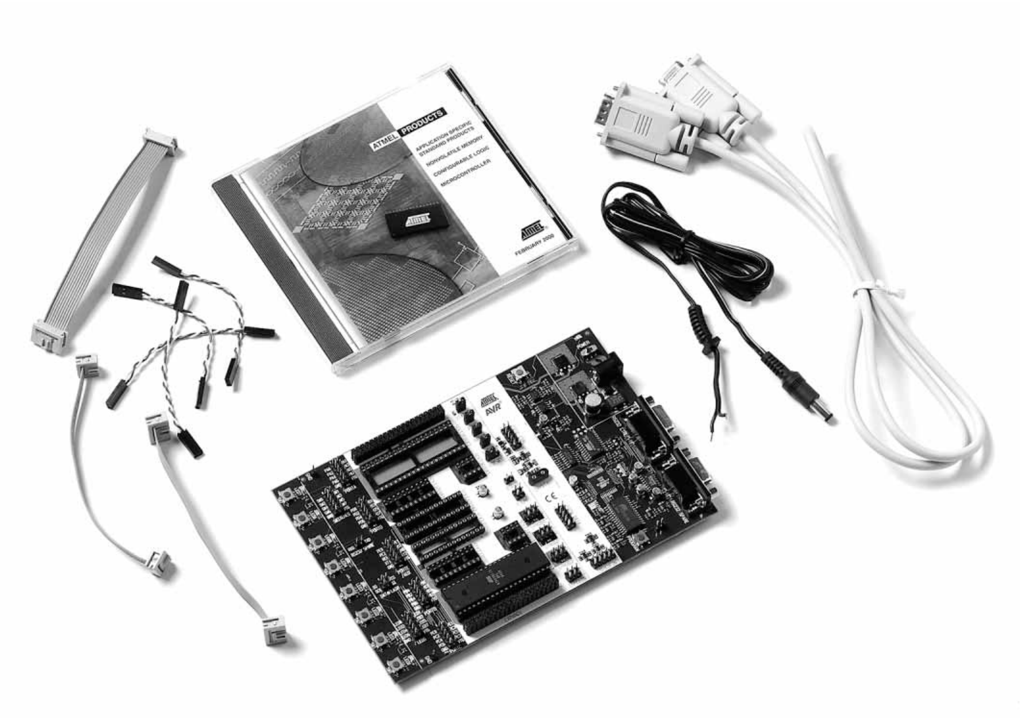
Congratulations on your purchase of the AVR® STK500 Flash Microcontroller Starter Kit. The STK500 is a complete starter kit and development system for the AVR Flash Microcontroller from Atmel Corporation. It is designed to give designers a quick start to develop code on the AVR and for prototyping and testing of new designs.

1.1 Starter Kit Features

- AVR Studio® Compatible
- RS-232 Interface to PC for Programming and Control
- Regulated Power Supply for 10 - 15V DC Power
- Sockets for 8-pin, 20-pin, 28-pin, and 40-pin AVR Devices
- Parallel and Serial High-voltage Programming of AVR Devices
- Serial In-System Programming (ISP) of AVR Devices
- In-System Programmer for Programming AVR Devices in External Target System
- Reprogramming of AVR Devices
- 8 Push Buttons for General Use
- 8 LEDs for General Use
- All AVR I/O Ports Easily Accessible through Pin Header Connectors
- Additional RS-232 Port for General Use
- Expansion Connectors for Plug-in Modules and Prototyping Area
- (NB! No longer valid: "On-board 2-Mbit DataFlash® for Nonvolatile Data Storage")

The STK500 is supported by AVR Studio, version 3.2 or higher. For up-to-date information on this and other AVR tool products, please read the document “avrtools.pdf”. The newest version of AVR Studio, “avrtools.pdf” and this user guide can be found in the AVR section of the Atmel web site, *www.atmel.com*.

Figure 1-1. STK500



1.2 Device Support

The system software currently supports the following devices in all speed grades:

- ATtiny11
- ATtiny12
- ATtiny15
- ATtiny22
- ATtiny28
- AT90S1200
- AT90S2313
- AT90S2323
- AT90S2333
- AT90S2343
- AT90S4414
- AT90S4433
- AT90S4434
- AT90S8515
- AT90S8535
- ATmega8
- ATmega16
- ATmega161
- ATmega163
- ATmega323
- ATmega103[1]
- ATmega128[1]

Note: ***1.*** In external target or in STK501, devices do not fit into the sockets of STK500.

Support for new AVR devices may be added in new versions of AVR Studio. The latest version of AVR Studio is always available from *www.atmel.com*.

Section 2

Getting Started

2.1 Unpacking the System

Kit contents:

- STK500 starter kit evaluation board
- Cables for STK500:
  - (2 pcs) 10-wire cables for I/O ports and parallel mode programming
  - (1 pc) 6-wire cable for In-System Programming
  - (4 pcs) 2-wire cable for UART and DataFlash connections
- 9-pin RS-232 cable
- DC power cable
- Atmel CD-ROM with datasheets and software
- AT90S8515-8PC sample microcontroller

2.2 System Requirements

The minimum hardware and software requirements are:

- 486 processor (Pentium® is recommended)
- 16 MB RAM
- 12 MB free hard disk space (AVR Studio)
- Windows® 95/98/2000/ME and Windows NT® 4.0 or higher
- 115200 baud RS-232 port (COM port)
- 10 - 15V DC power supply, 500 mA min.

2.3 Quick Start

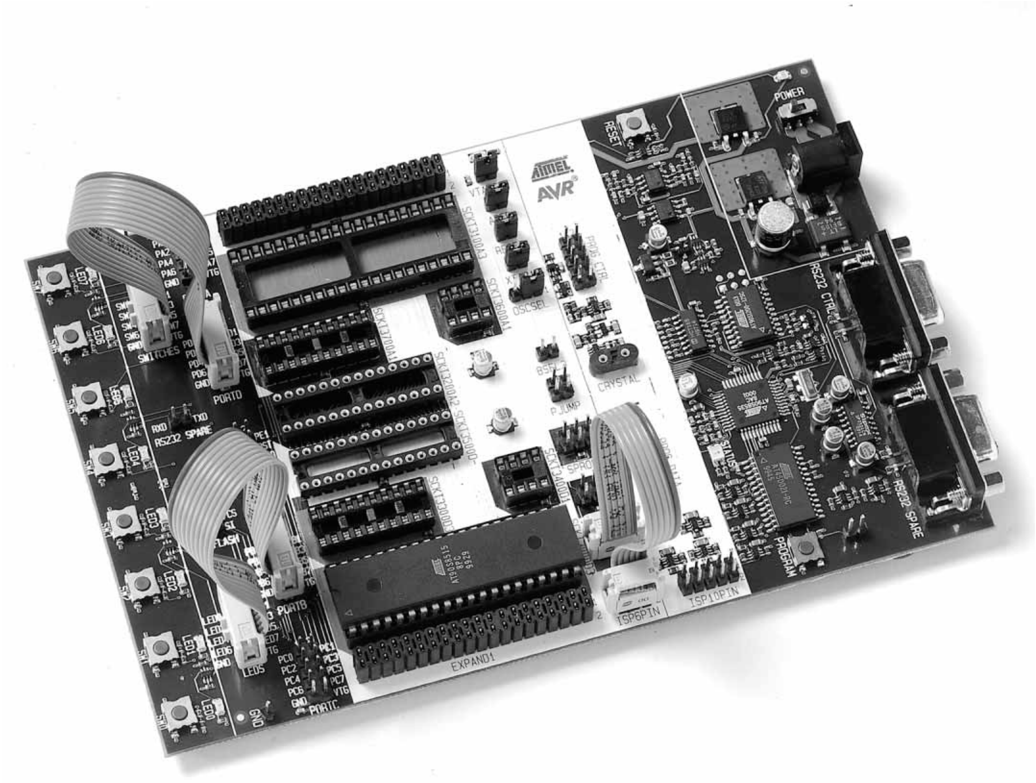
The STK500 starter kit is shipped with an AT90S8515-8PC microcontroller in the socket marked SCKT3000D3. The default jumper settings will allow the microcontroller to execute from the clock source and voltage regulator on the STK500 board.

The microcontroller is programmed with a test program that toggles the LEDs. The test program in the AT90S8515 is similar to the example application code described in Section 9. Connect the LEDs and switches and power up the STK500 to run the test program in the AT90S8515.

Use the supplied 10-pin cables to connect the header marked “PORTB” with the header marked “LEDS”, and connect the header marked “PORTD” with the header marked “SWITCHES”. The connections are shown in Figure 2-1.

An external 10 - 15V DC power supply is required. The input circuit is a full bridge rectifier, and the STK500 automatically handles both positive or negative center connectors. If a positive center connector is used, it can be impossible to turn the STK500 off since the power switch disconnects the GND terminal. In this case, GND can be supplied through the RS-232 cable shield if connected or through alternative GND connections. Connect the power cable between a power supply and the STK500. Apply 10 - 15V DC to the power connector. The power switch turns the STK500 main power on and off. The red LED is lit when power is on, and the status LEDs will go from red, via yellow, to green. The green LED indicates that the target V_{CC} is present. The program now running in the AT90S8515 will respond to pressed switches by toggling the LEDs.

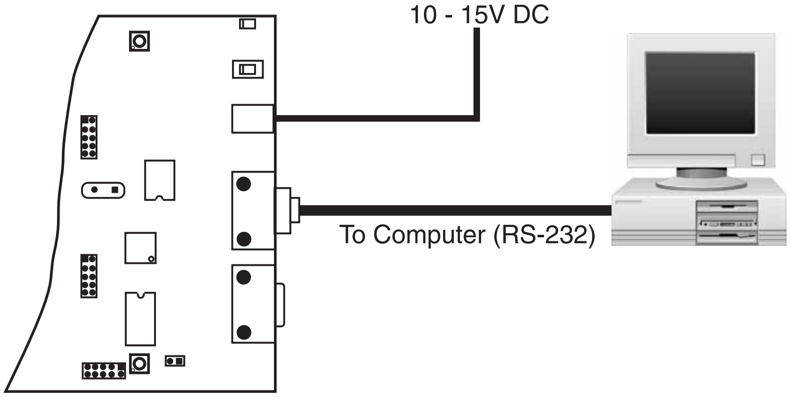
Figure 2-1. Default Setup of STK500



The starter kit can be configured for various clock and power sources. A complete description of the jumper settings is explained in paragraph 3.8 on page 3-15 and on the reverse side of the starter kit.

2.3.1 Connecting the Hardware

Figure 2-2. Connection to STK500



To program the AT90S8515, connect the supplied 6-wire cable between the ISP6PIN header and the SPROG3 target ISP header as shown in Figure 2-1. Section 3.7.1 on page 3-9 describes the programming cable connections.

Connect a serial cable to the connector marked “RS232 CTRL” on the evaluation board to a COM port on the PC as shown in Figure 2-2. Install AVR Studio software on the PC. Instructions on how to install and use AVR Studio are given in Section 5 on page 5-1. When AVR Studio is started, the program will automatically detect to which COM port the STK500 is connected.

2.3.2 Programming the Target AVR Device

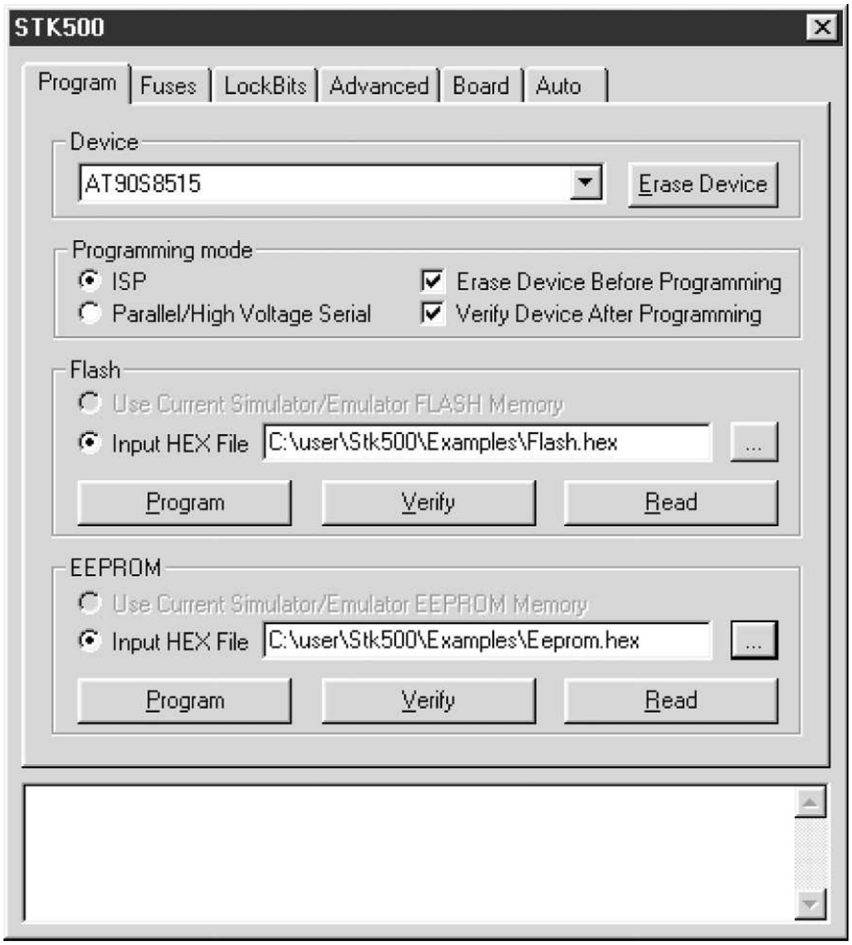
The STK500 is controlled from AVR Studio, version 3.2 and higher. AVR Studio is an integrated development environment (IDE) for developing and debugging AVR applications. AVR Studio provides a project management tool, source file editor, simulator, in-circuit emulator interface and programming interface for STK500.

To program a hex file into the target AVR device, select “STK500” from the “Tools” menu in AVR Studio.

Select the AVR target device from the pull-down menu on the “Program” tab and locate the intel-hex file to download.

Press the “Erase” button, followed by the “Program” button. The status LED will now turn yellow while the part is programmed, and when programming succeeds, the LED will turn green. If programming fails, the LED will turn red after programming. See the troubleshooting guide in Section 7 on page 7-1.

Figure 2-3. AVR Studio STK500 Programming Menu



Device

AT90S8515

Programming mode

● ISP

○ Parallel/High Voltage Serial

☑ Erase Device Before Programming

☑ Verify Device After Programming

Flash

○ Use Current Simulator/Emulator FLASH Memory

● Input HEX File C:\user\Stk500\Examples\Flash.hex

EEPROM

○ Use Current Simulator/Emulator EEPROM Memory

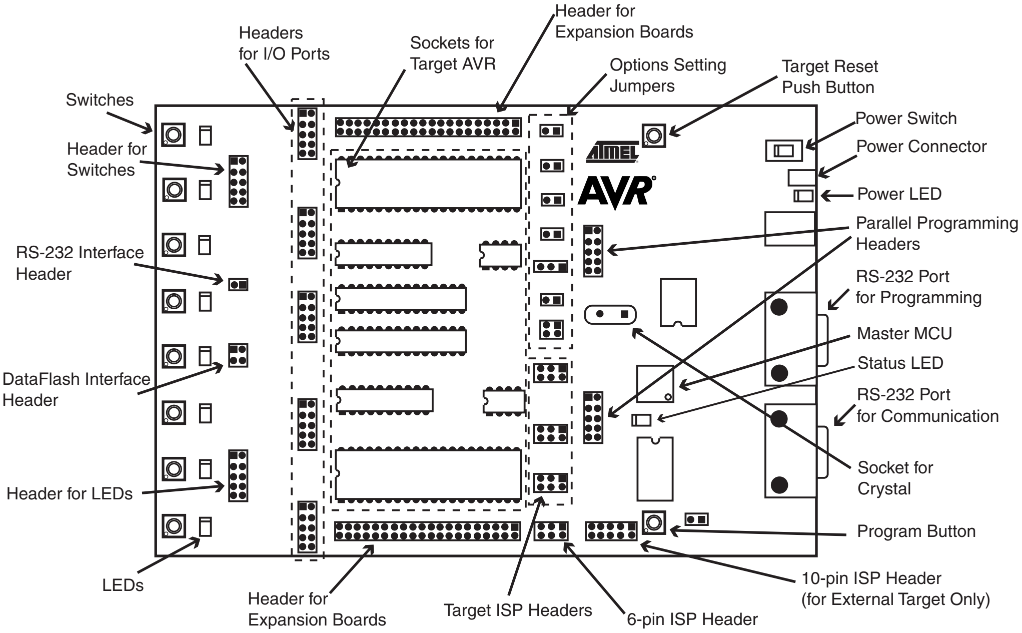
● Input HEX File C:\user\Stk500\Examples\Eeprom.hex

Complete descriptions of using the STK500 interface in AVR Studio are given in Section 5 on page 5-1.

Section 3

Hardware Description

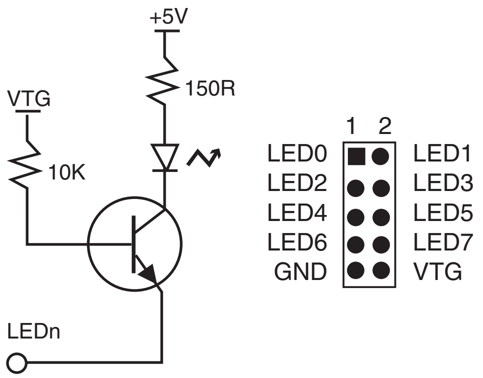
Figure 3-1. STK500 Components



3.1 Description of User LEDs

The STK500 starter kit includes 8 yellow LEDs and 8 push-button switches. The LEDs and switches are connected to debug headers that are separated from the rest of the board. They can be connected to the AVR devices with the supplied 10-wire cable to the pin header of the AVR I/O ports. Figure 3-4 shows how the LEDs and switches can be connected to the I/O port headers. The cables should be connected directly from the port header to the LED or switch header. The cable should not be twisted. A red wire on the cable indicates pin 1. Confirm that this is connected to pin 1 on each of the headers. Figure 3-2 shows how the LED control is implemented. This solution will give the same amount of light from the LED for all target voltages from 1.8V to 6.0V.

Figure 3-2. Implementation of LEDs and LED Headers



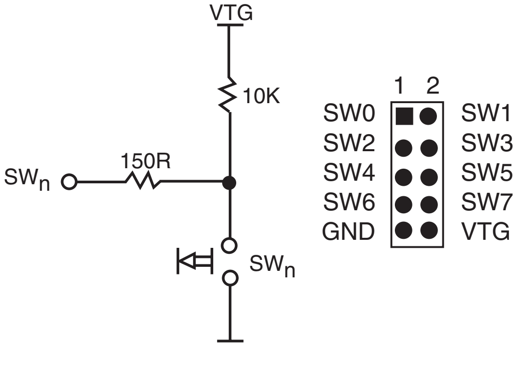
Note: The AVR can source or sink enough current to drive a LED directly. In the STK500 design, a transistor with two resistors is used to give the same amount of light from the LED, whatever the target voltage (VTG) may be and to turn off the LEDs when VTG is missing.

3.2 Description of User Switches

The switches connected to the debug headers are implemented as shown in Figure 3-3.

Pushing a switch causes the corresponding SWx to be pulled low, while releasing it will result in VTG on the appropriate switch header connector. Valid target voltage range is 1.8V < VTG < 6.0V.

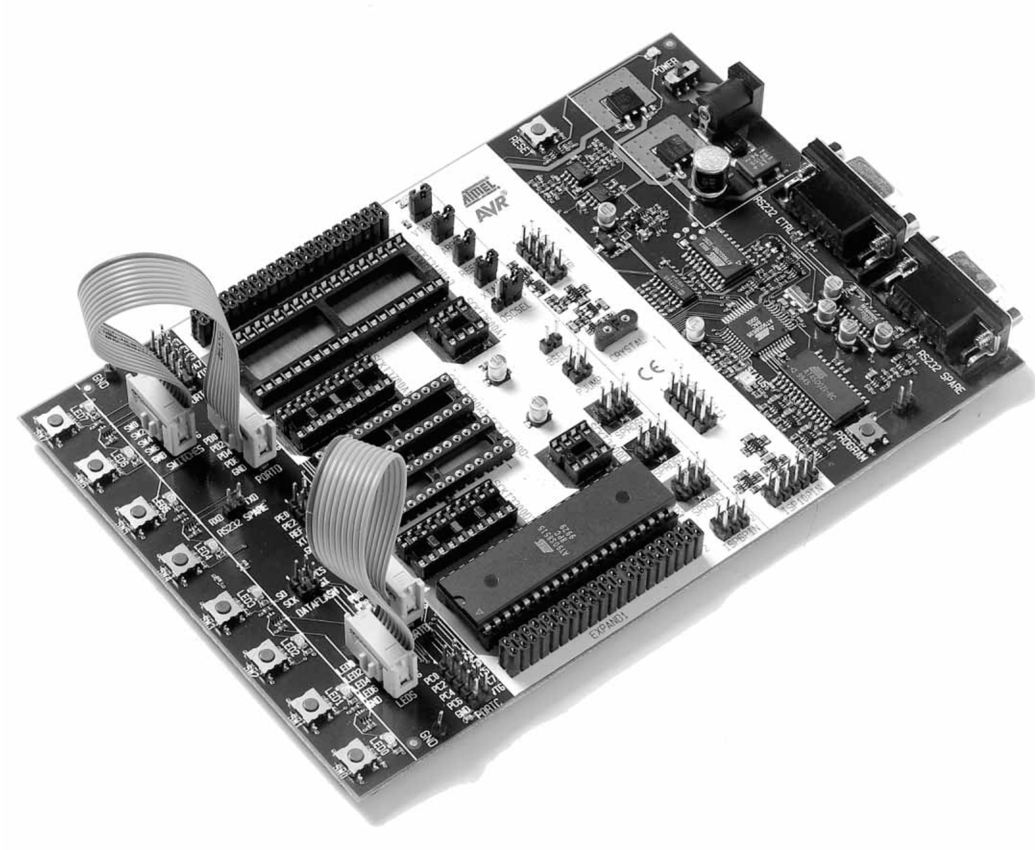
Figure 3-3. Implementation of Switches and Switch Headers



Note: In the AVR, the user can enable internal pull-ups on the input pins, removing the need for an external pull-up on the push-button. In the STK500 design, we have added an external 10K pull-up to give all users a logical “1” on SWn when the push-button is not pressed. The 150R resistor limits the current going into the AVR.

3.3 Connection of LEDs and Switches

Figure 3-4. Connection of LEDs and Switches to I/O Port Headers

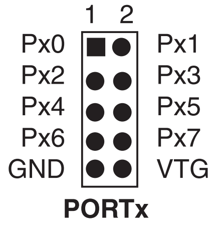


Any I/O port of the AVR can be connected to the LEDs and switches using the 10-wire cables. The headers are supplied with VTG (target V_{CC}) and GND lines in addition to the signal lines.

3.4 Port Connectors

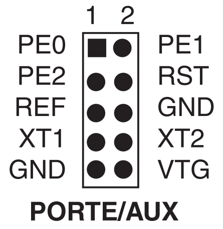
The pinout for the I/O port headers is explained in Figure 3-5. The square marking indicates pin 1.

Figure 3-5. General Pinout of I/O Port Headers



The PORTE/AUX header has some special signals and functions in addition to the PORTE pins. The pinout of this header is shown in Figure 3-6.

Figure 3-6. Pinout of PORTE Header



The special functions of this port are:

- **PE0 - PE2:**

Table 3-1. PORTE Connection

	ATmega161	**AT90S4414/AT90S8515**
PE0	PE0/ICP/INT2	ICP
PE1	PE1/ALE	ALE
PE2	PE2/OC1B	OC1B

- **REF:** Analog reference voltage. This pin is connected to the AREF pin on devices having a separate analog reference pin.
- **XT1:** XTAL 1 pin. The internal main clock signal to all sockets. If the XTAL1 jumper is disconnected, this pin can be used as external clock signal.
- **XT2:** XTAL 2 pin. If the XTAL1 jumper is disconnected, this pin can be used for external crystal with the XT1 pin.

The headers for the LEDs and switches use the same pinout as the I/O port headers. The pinout of the switch header is explained in Figure 3-7 and the pinout for the LED header is explained in Figure 3-8. The square marking indicates pin 1.

Figure 3-7. Pinout of the Switch Header

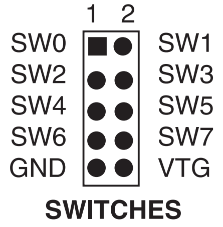
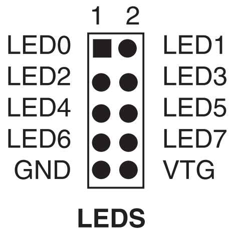


Figure 3-8. Pinout of the LED Header



3.5 Description of User RS-232 Interface

The STK500 includes two RS-232 ports. One RS-232 port is used for communicating with AVR Studio. The other RS-232 can be used for communicating between the target AVR microcontroller in the socket and a PC serial port connected to the RS-232. To use the RS-232, the UART pins of the AVR need to be physically connected to the RS-232.

The 2-pin header marked “RS232 SPARE” can be used for connecting the RS-232 converter to the UART pins on the target AVR microcontroller in the socket. Use the 2-wire cable to connect the UART pins to the RS-232. The connection is shown in Figure 3-9. The block schematic of the RS-232 connection is shown in Figure 3-10.

Figure 3-9. Connection of I/O Pins to UART

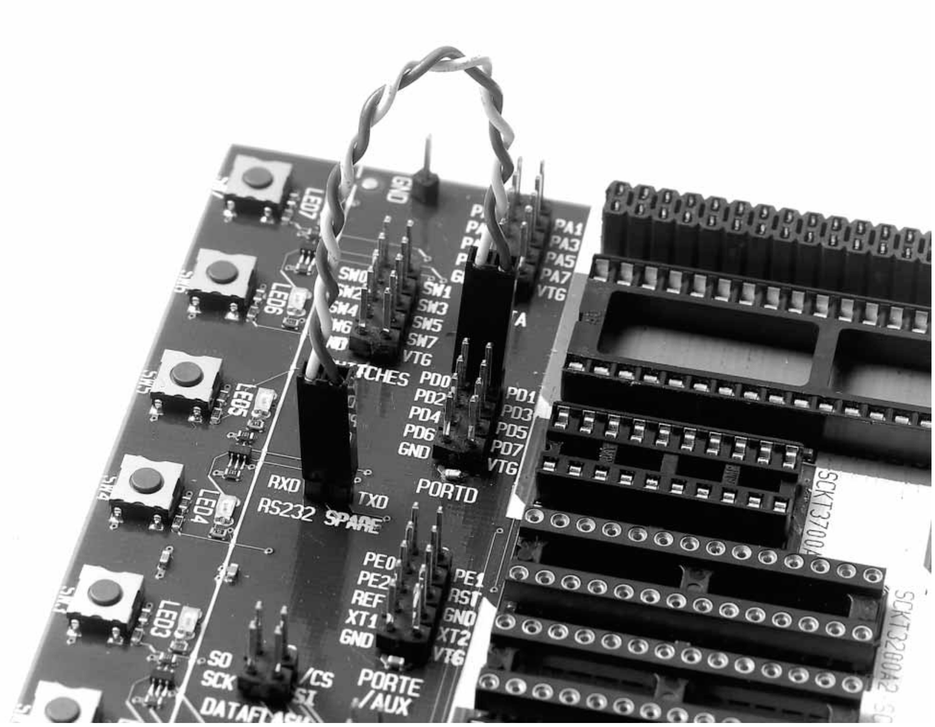
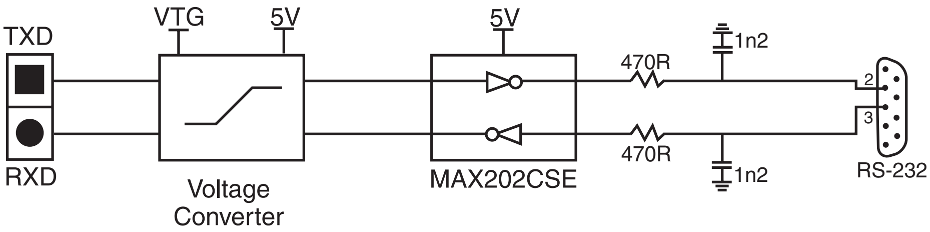


Figure 3-10. Schematic of UART Pin Connections



3.6 Description of DataFlash Pins

NB! Not valid: AT45D021 2-Mbit DataFlash is included on the STK500 for data storage. A DataFlash is a high-density Flash memory with SPI serial interface. A detailed datasheet of the DataFlash can be obtained from the Flash memory section of the Atmel CD-ROM or from the Atmel web site.

The DataFlash can be connected to the I/O pins of the microcontroller sockets. The 4-pin header marked “DATAFLASH” can be used for connecting the SPI interface of the DataFlash to the I/O pins on the target AVR microcontroller in the socket. 2-wire cables are included with STK500 for connecting the DataFlash to the I/O pins. The supplied 10-wire cables can also be used if the DataFlash is connected to the hardware SPI interface on PORTB of the AVR microcontroller. The connection of the I/O pins is shown in Figure 3-13. The block schematic of the DataFlash connection is shown in Figure 3-14, for connection of the DataFlash to the AVR hardware SPI interface. The SPI interface pinout is shown in Figure 3-11 and Figure 3-12.

Figure 3-11. PORTB SPI Pinout (40-pin Parts)

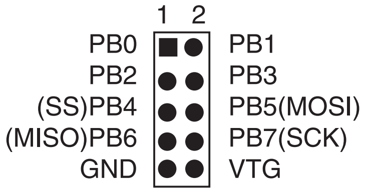


Figure 3-12. PORTB SPI Pinout (28-pin Analog Parts)

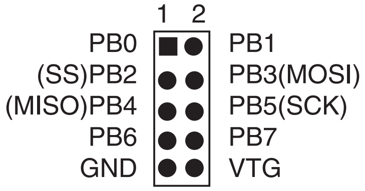


Figure 3-13. Connection of I/O Pins to DataFlash for AT90S8515

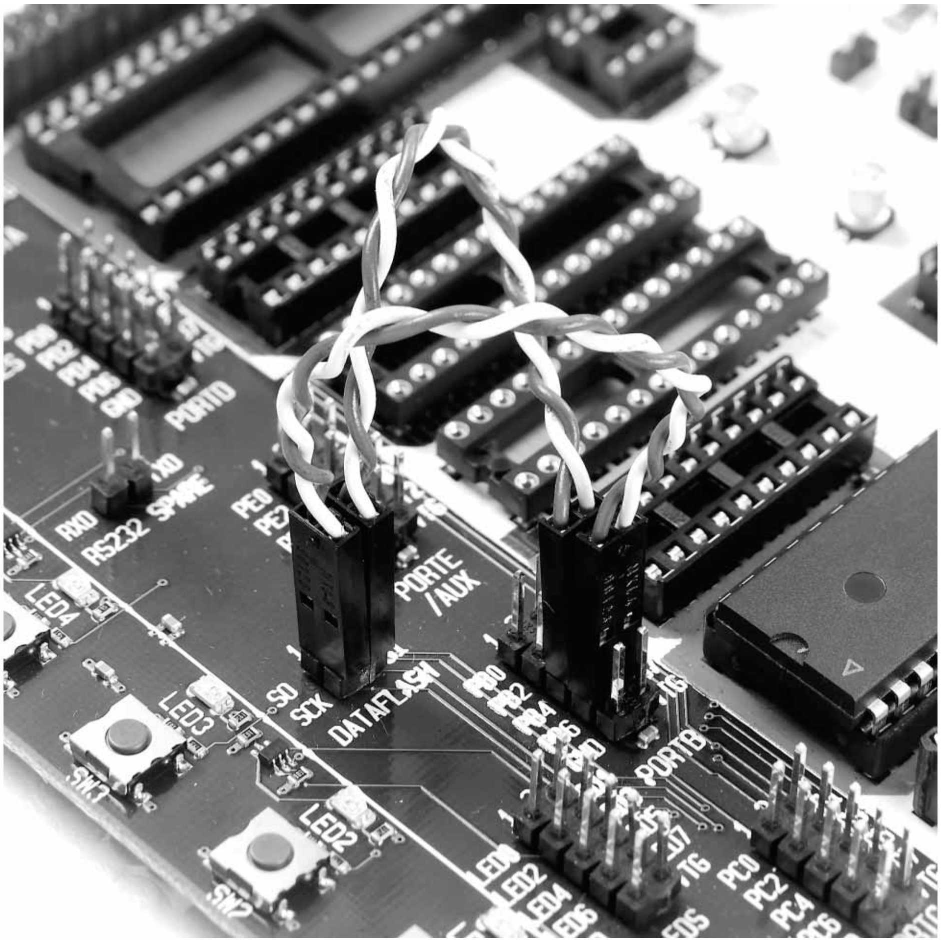
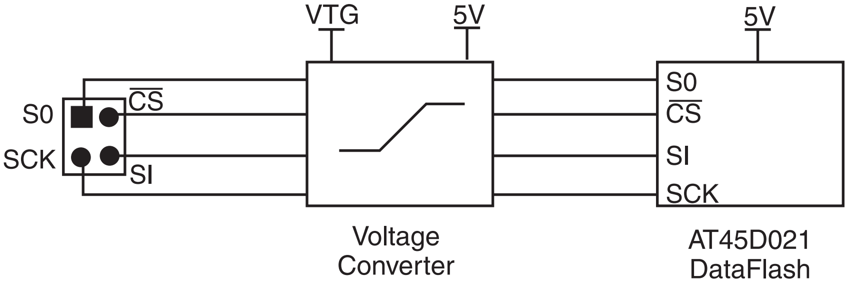


Figure 3-14. Schematic of DataFlash Connections



3.7 Target Socket Section

The programming module consists of the eight sockets in the white area in the middle of the starter kit. In these sockets, the target AVR devices can be inserted for programming and are used in the application.

Note: Only one AVR device should be inserted in the sockets at a time.

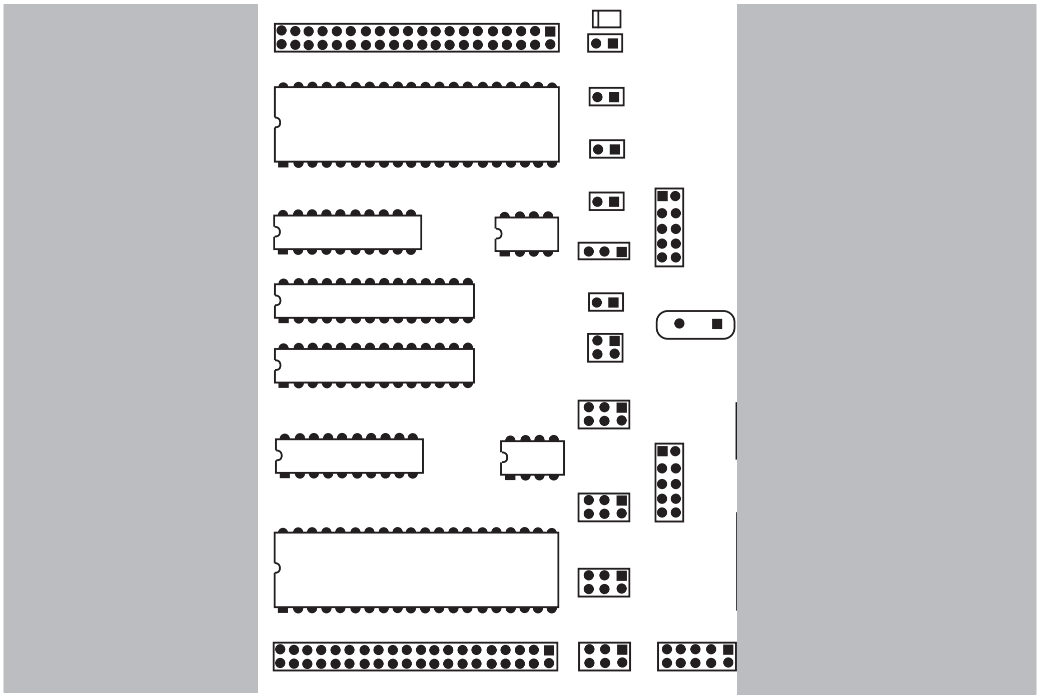
The AVR Flash memory is guaranteed to be correct after 1,000 programming operations; the typical lifetime of the Flash memory is much longer.

Note: When inserting a device in the socket, notice the orientation of the device.

The notch on the short side of the part must match the notch on the socket. If the device is inserted the wrong way, it may damage the part and the starter kit.

The socket section is used for both running applications and target device programming.

Figure 3-15. The STK500 Programming Module



The part inserted in the socket can be programmed in the system from AVR Studio with two different methods:

1. AVR In-System Programming (ISP) running at the parts normal supply voltage.
2. High-voltage Programming, where the supply voltage is always 5 volts.

Four general nets (VTARGET, RESET, XTAL1 and AREF) can be connected to the socket section.

The following sections describe how to use both programming methods. For instructions on using the AVR Studio programming software, see Section 5, “Using AVR Studio” on page 5-1.

3.7.1 ISP Programming

In-System Programming uses the AVR internal SPI (Serial Peripheral Interface) to download code into the Flash and EEPROM memory of the AVR. ISP programming requires only V_{CC}, GND, RESET and three signal lines for programming. All AVR devices except AT90C8534, ATtiny11 and ATtiny28 can be ISP programmed. The AVR can be programmed at the normal operating voltage, normally 2.7 - 6.0V. No high-voltage signals are required. The ISP programmer can program both the internal Flash and EEPROM. It also programs fuse bits for selecting clock options, start-up time and internal Brown-out Detector (BOD) for most devices.

High-voltage programming can also program devices that are not supported by ISP programming. Some devices require High-voltage Programming for programming certain fuse bits. See the High-voltage Programming section on page 3-11 for instructions on how to use High-voltage Programming.

Because the programming interface is placed on different pins from part to part, three programming headers are used to route the programming signals to the correct pins. A 6-wire cable is supplied for connecting the ISP signals to the target ISP header. A color coding system and a number system are used to explain which target ISP header is used for each socket.

During ISP programming, the 6-wire cable must always be connected to the header marked “ISP6PIN”. When programming parts in the blue sockets, connect the other end of the cable to the blue SPROG1 target ISP header. When programming parts in the green socket, use the green SPROG2 target ISP header. And when programming parts in the red sockets, use the red SPROG3 target ISP header.Table 3-2 shows which socket suits which AVR device, and which SPROG target ISP header to use for ISP programming.

The 6-wire cables should be connected directly from the ISP6PIN header to the correct SPROG target ISP header. The cable should not be twisted. A colored wire on the cable indicates pin 1. Confirm that this is connected to pin 1 on each of the headers.

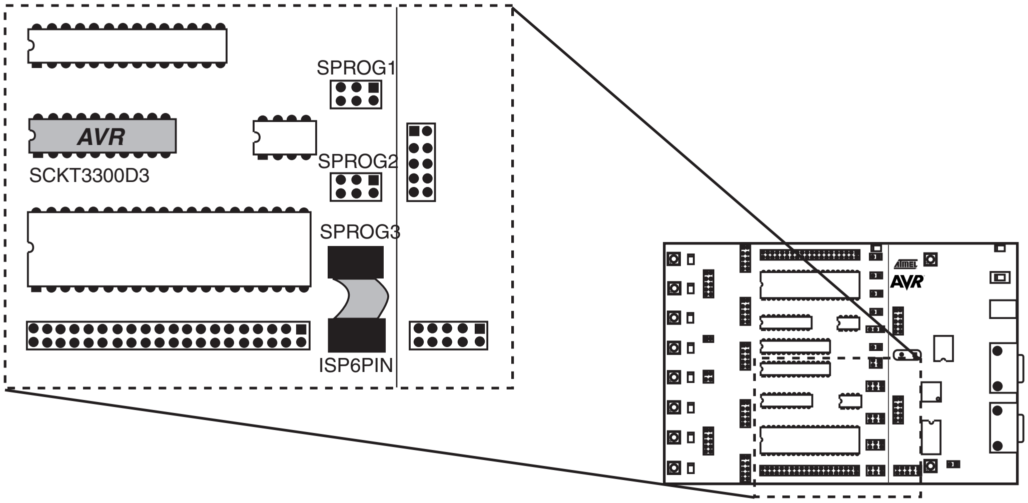
When programming 8-pin devices, note the following: Pin 1 is used both as RESET and as PB5 on some devices (ATtiny11, ATtiny12 and ATtiny15). Pin 1 on the 8-pin sockets SCKT3400D1 and SCKT3400D1 are connected to PB5. The RESET signal used during ISP programming is therefore not connected to pin 1 on these sockets. This signal must be connected by placing a wire between RST and the PORTE header and PB5 on the PORTB header.

Table 3-2. AVR Sockets

AVR Devices	STK500 Socket	Color	Number	Target ISP Header
AT90S1200 AT90S2313	SCKT3300D3	Red	3	SPROG3
AT90S2323 AT90S2343 ATtiny12 ATtiny22	SCKT3400D1	Blue	1	SPROG1. Connect RST on PORTE to PB5 on PORTB. Connect XTI on PORTE to PB3 (XTAL1 on 2323) on PORTB.
ATtiny11	SCKT3400D1	Blue	1	High-voltage Programming only
ATtiny28	SCKT3500D-	None	–	High-voltage Programming only
AT90S4414 AT90S8515 ATmega161	SCKT3000D3	Red	3	SPROG3
AT90S4434 AT90S8535 ATmega16 ATmega163 ATmega323	SCKT3100A3	Red	3	SPROG3
AT90S2333 AT90S4433 ATmega8	SCKT3200A2	Green	2	SPROG2
ATtiny15	SCKT3600A1	Blue	1	SPROG1. Connect RST on PORTE to PB5 on PORTB.
N/A	SCKT3700A1	Blue	1	Socket is not in use in this version of STK500
ATmega103 ATmega128	Use the STK501 Top Module			

Figure 3-16 shows an example of how AT90S2313 can be In-System Programmed. The 6-wire cable is connected from the ISP6PIN header to the red SPROG3 target ISP header, and the AT90S2313 part is inserted in the red socket marked “SCKT3100D3”.

Figure 3-16. Example Connection for Programming AT90S2313



It is not necessary to remove the 6-wire cable from its ISP position while running a program in the AVR. The port pins used for ISP programming can be used for other purposes in your program.

3.7.2 High-voltage Programming

For High-voltage Programming, a 12V programming voltage is applied to the RESET pin of the AVR device. All AVR devices can be programmed with High-voltage Programming, and the target device can be programmed while it is mounted in its socket.

Two different methods are used for High-voltage Programming: 8-pin parts use a serial programming interface, while other parts use a parallel programming interface. The programming signals are routed to the correct pins of the target device using the cables supplied with STK500.

Table 3-3 summarizes the programming method and special considerations when using High-voltage Programming.

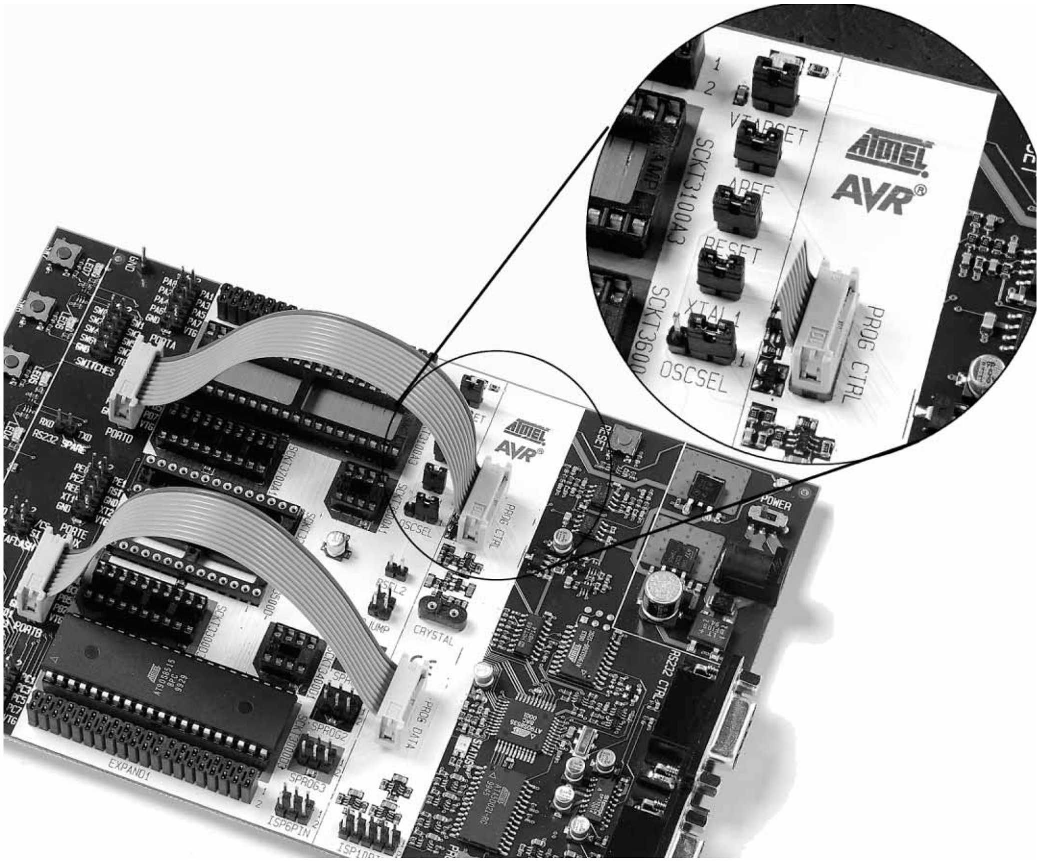
Table 3-3. High-voltage Programming Settings

AVR Devices	STK500 Socket	Color	Number	High-voltage Programming Method
AT90S1200 AT90S2313	SCKT3300D3	Red	3	Parallel High-voltage Programming. Connect PROG CTRL header to PORTD and PROG DATA to PORTB, as shown in Figure 3-17 on page 3-12.
AT90S4414 AT90S8515	SCKT3000D3	Red	3	
AT90S4434 AT90S8535	SCKT3100A3	Red	3	
ATtiny28	SCKT3500D-	None	–	
ATmega161	SCKT3000D3	Red	3	Parallel programming as above; mount BSEL2 jumper. See Section 3.8.
ATmega16 ATmega163 ATmega323	SCKT3100A3	Red	3	
AT90S2333 AT90S4433	SCKT3200A2	Green	2	Parallel programming as above; mount PJUMP jumpers. See Section 3.8.
ATmega103 ATmega128	Use the STK501 Top Module	–	–	Parallel programming as above; mount BSEL2 jumper. See Section 3.8.
ATmega8	SCKT3200A2	Green	2	Parallel programming as above; Mount PJUMP jumpers and mount BSEL2 terminal to PC2. See Section 3.8.5 and Section 3.8.6.
AT90S2323 AT90S2343 ATtiny11 ATtiny12 ATtiny22	SCKT3400D1	Blue	1	Serial High-voltage Programming
ATtiny15	SCKT3600A1	Blue	1	
N/A	SCKT3700A1	Blue	1	Socket not in use in this version of STK500

3.7.2.1 Parallel High-voltage Programming

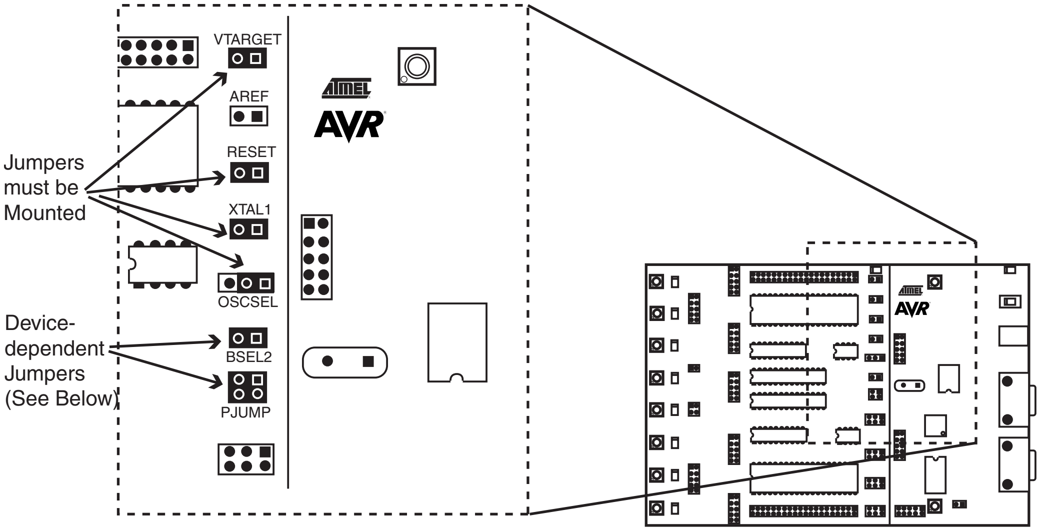
To use High-voltage Programming, the programming signal must be routed to the AVR I/O pins. The two 10-wire cables supplied with the STK500 can be used to connect the PROG DATA header to the PORTB header and the PROG CTRL header to the PORTD header, as shown in Figure 3-17.

Figure 3-17. Connection for Parallel High-voltage Programming



Some of the jumper settings on STK500 must be changed when using High-voltage Programming. Figure 3-18 explains these jumper settings.

Figure 3-18. Jumper Settings for High-voltage Programming



Hardware setup for parallel High-voltage Programming:

1. Switch power off.
2. Place the device to program in its socket according to Table 3-3 on page 3-11.
3. Connect the headers PROGDATA and PORTB with the 10-wire cable.
4. Connect the headers PROGCTRL and PORTD with the 10-wire cable.
5. Mount jumper OSCSEL on pins 1 and 2 to select software-controlled clock.
6. Mount jumper XTAL1 to route the oscillator signal to the device.
7. Mount jumpers VTARGET and RESET.
8. When programming AT90S2333, AT90S4433, or ATmega8, mount both PJUMP jumpers. The 2-wire cables can be used instead of jumpers.
9. When programming ATmega16, ATmega163, ATmega161, ATmega128, or ATmega323, mount the BSEL2 jumper. When programming ATmega8, connect BSEL2 terminal to PC2. A 2-wire cable can be used instead of jumpers.
10. Disconnect target system.
11. Switch power on.
12. Ensure that VTARGET is between 4.5V and 5.5V before programming. See Section 5.3.5.1.

For a complete description of jumper settings, see Section 3.8, “Jumper Settings”.

Note: Remove the hardware setup for High-voltage Programming before starting a debug session.

3.7.2.2 Serial High-voltage Programming

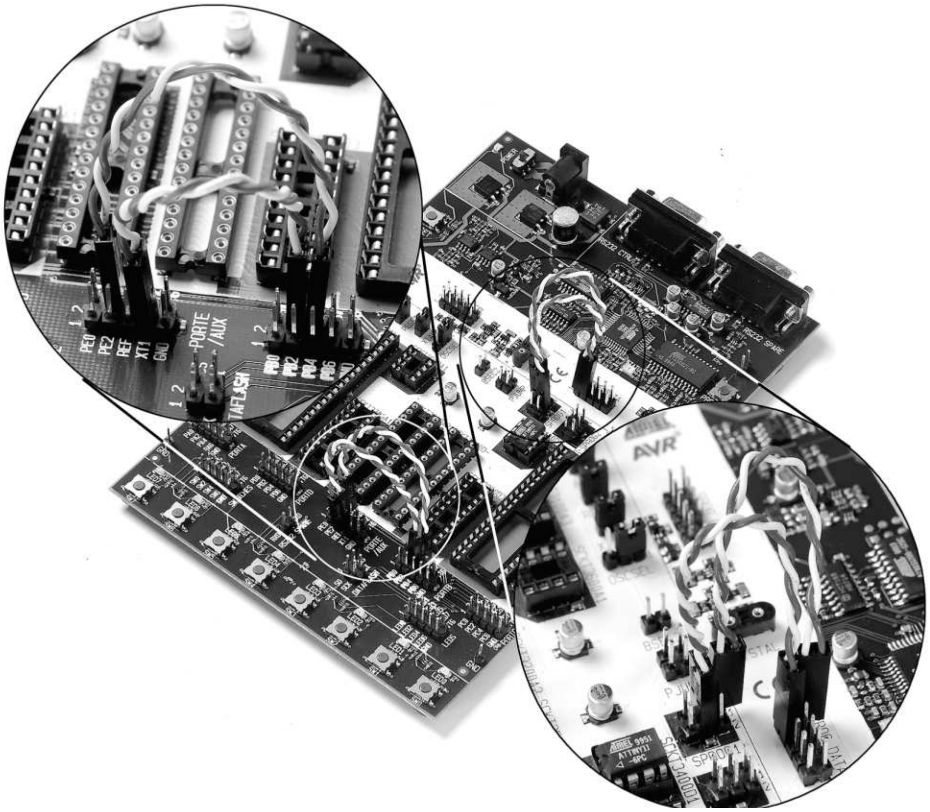
The 8-pin AVRs have too few pins to use parallel communication during High-voltage Programming. They use serial communication instead.

This means that fewer signals have to be routed. Hardware setup for serial High-voltage Programming is as follows:

1. Switch power off.
2. Place the device to program in its socket according to Table 3-3 on page 3-11.
3. Mount jumper OSCSEL on pins 1 and 2 to select software-controlled clock.
4. Mount jumper XTAL1 to route the oscillator signal to the device.
5. Mount jumpers VTARGET and RESET.
6. Use one 2-wire cable to connect the PB3 pin (pin 4) on the PORTB header to the XT1 pin (pin 7) on the PORTE/AUX header. This will connect the clock system to the AVR device.
7. Use another 2-wire cable to connect the PB5 pin (pin 6) on the PORTB header to the RST pin (pin 4) on the PORTE/AUX header. This will connect the reset system to the AVR device.
8. Use a third 2-wire cable to connect the PB0 and PB2 pins (pins 4 and 3) on the SPROG1 header to the DATA0 and DATA2 pins (pins 1 and 3) on the PROG DATA header.
9. Use the last 2-wire cable to connect the PB1 pin (pin 1) on the SPROG1 header to the DATA1 pin (pin 2) on the PROG DATA header.
10. Switch power on and you are ready to program.

All connections are shown in Figure 3-19.

Figure 3-19. Connection for Serial High-voltage Programming



3.8 Jumper Settings

A master microcontroller and the eight jumpers control the hardware settings of the starter kit. During normal operation these jumpers should be mounted in the default position. To configure the starter kit for advanced use, the jumpers can be removed or set to new positions. The jumper settings and usage are explained in the following section. The default setting of the jumpers are shown in Figure 3-20.

Figure 3-20. Default Jumper Setting

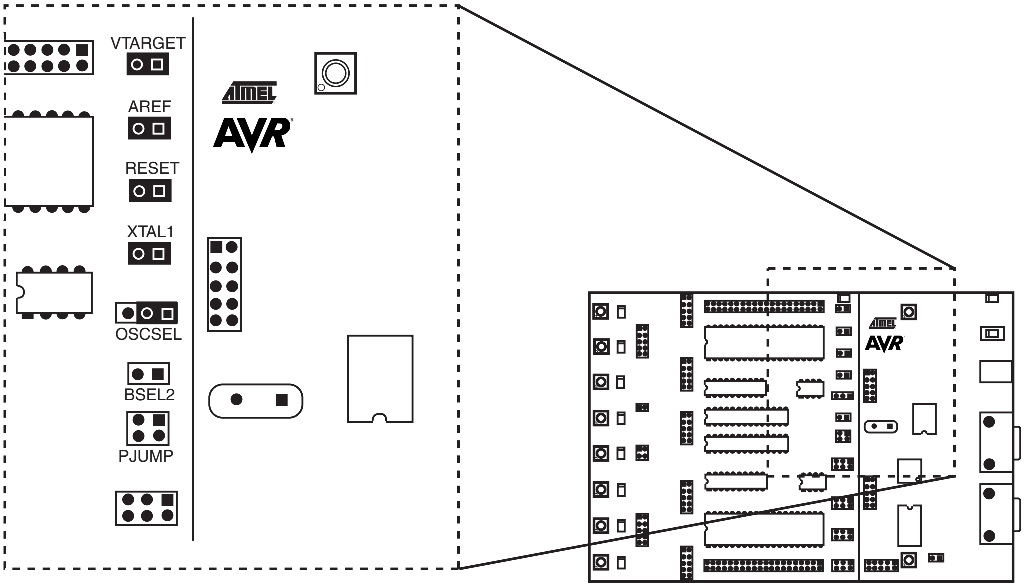


Table 3-4. Description of Jumpers

Jumper	Description of Default Setting
VTARGET	On-board VTARGET supply connected
AREF	On-board Analog Voltage Reference connected
RESET	On-board Reset System connected
XTAL1	On-board Clock System connected
OSCSEL	On-board Oscillator selected
BSEL2	Unmounted. Used for High-voltage Programming of ATmega8, ATmega16, ATmega161, ATmega163, ATmega128, and ATmega323
PJUMP	Unmounted. Used for High-voltage Programming of AT90S2333, AT90S4433, and ATmega8