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

Contact us

Programmers and Debuggers

AVR Dragon

USER GUIDE

The Atmel AVR Dragon Debugger

With the Atmel® AVR® Dragon, Atmel has set a new standard for low-cost development tools. AVR Dragon™ supports all programming modes for the Atmel AVR device families. It also includes full debugging support for most Atmel AVR devices.

Table of Contents

1. Introducing AVR Dragon

Figure 1-1. Front side

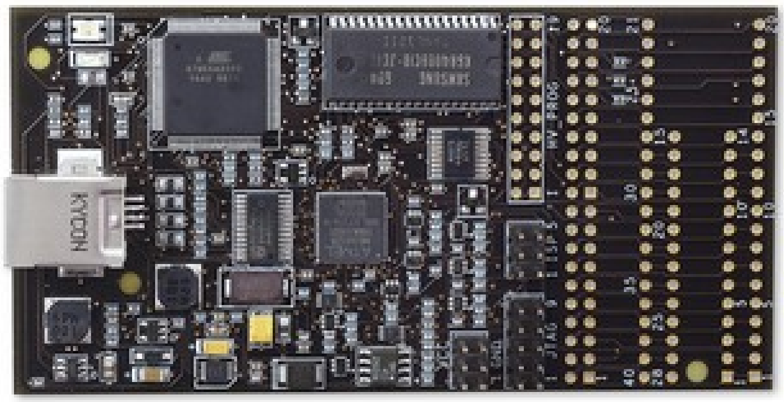
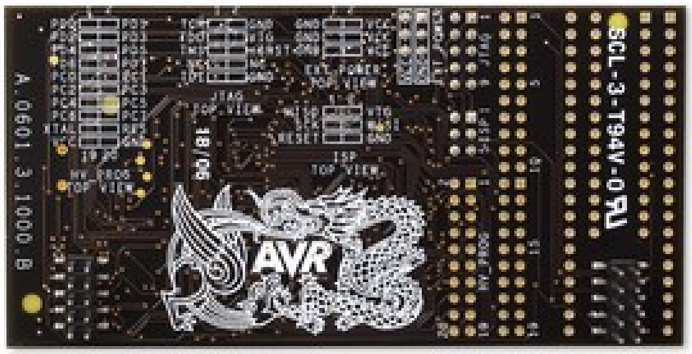


Figure 1-2. Back side



With the **Atmel AVR Dragon**, Atmel has set a new standard for low-cost development tools. AVR Dragon supports all programming modes for the Atmel AVR device families. It also includes full debugging support for most AVR devices.

At a fraction of the price traditionally associated with this kind of featured tool, the AVR Dragon will fulfill all your programming and debugging needs. The flexible and secure firmware upgrade feature allows the software front-end to easily upgrade the AVR Dragon to support new devices.

To see which devices are currently supported read the Atmel Studio release notes/readme.

New devices will be added through Atmel Studio updates on a regular basis.

1.1. Supported Protocols

Currently the following protocols are supported:

1.1.1. Programming Interfaces

- SPI programming (SPI)

- High Voltage Serial Programming (HVSP)
- Parallel Programming (PP)
- JTAG Programming (JTAG)
- PDI Programming (PDI)
- aWire Programming (aWire)

1.1.2. Debugging Interfaces

- JTAG (JTAG)
- debugWIRE (dW)
- PDI (PDI)
- aWire (aWire)

1.2. Overview

Atmel AVR Dragon can be used with an external target board. However, the onboard prototype area allows simple programming and debugging without any additional hardware besides strapping cables. See the Using the Atmel AVR Prototype Area section for a description on how to use this.

AVR Dragon is powered by the USB cable, and can also source an external target with up to 300mA (from the VCC connector, 5V) when programming or debugging. For more information on technical details, read the AVR Dragon Requirements section. If the target is already powered by an external power source, the AVR Dragon will adapt and level convert all signals between the target and the AVR Dragon.

Note:
If the target board is powered by an external power source, no connection should be made between the VCC connector and the external board.

AVR Dragon is fully supported by Atmel Studio (hereafter called the software front-end). This allows the AVR Dragon firmware to be easily updated to support new devices and protocols. When connecting the AVR Dragon, the software front-end will automatically check the firmware and prompt the user if an updated firmware is available.

2. Known Issues

- JTAG communication with packages in PDIP which have the CKOUT fuse enabled and running above 3.5V may be unstable if there is a long wiring from the Atmel AVR Dragon to the PDIP AVR.
- **High voltage programming issue, all targets:** Parallel Programming and High Voltage Serial Programming might not work if the startup time is set to 0ms (SUT fuses). The problem gets worse if the CKDIV8 fuse is not set.
- **ATtiny84 Programming issue:** Parallel Programming may fail on the ATtiny84 if both the DWEN fuse and any of the external clock fuses are enabled at the same time. A workaround is to use the Atmel STK®500 platform for Parallel Programming of this part.
- **ATtiny26 Programming issue:** Parallel Programming on ATtiny26 is unstable with the AVR Dragon. A workaround is to use the STK500 platform for Parallel Programming of this part.
- In order to set SPI frequency, AVR Dragon needs to sense target voltage. See the troubleshooting guide.
- **XMEGA® PDI issues:** XMEGA PDI mode on AVR Dragon does NOT work for the following XMEGA devices: A3/D3 - revisions B, C, and E, or A1 (up to revision K).

3. Getting Started

3.1. Before Starting

Important:
Read this section before connecting the Atmel AVR Dragon to the computer or target.

Important:
Install the Atmel Studio software front-end and the USB driver before connecting AVR Dragon to your PC.

Follow these simple steps to get started using the AVR Dragon:

1. Download the latest version of Atmel Studio.
2. Install the software front-end and the USB driver.
3. Connect the AVR Dragon to the computer, and auto-install new hardware (AVR Dragon) on the computer.
4. Start the software front-end.
5. Connect AVR Dragon to the target.

3.1.1. USB Setup

In order to use the Atmel AVR Dragon it is required to install the USB driver first (comes with the software front-end). Do not connect the AVR Dragon to the computer before running the USB Setup in order to follow the procedures described in Software and USB Setup.

3.1.2. Unpacking the Atmel AVR Dragon

The box contains:

- Atmel AVR Dragon tool
- Internet link to Software (http://www.atmel.com/avrdragon)

There is no CD shipped with the AVR Dragon. The only way of getting the software is by downloading it directly from the Internet.

You will also need: (not included)

- PC with free USB connector or a USB HUB capable of delivering 500mA
- USB Cable
- Latest Atmel Studio - minimal requirement 4.12 SP2 (Link: http://www.atmel.com/avrdragon)
- 6/10 pin Header Connector (or similar cables to connect the AVR Dragon to the target board)

3.1.3. System Requirements

The minimum hardware and software requirements are:

1. Pentium (Pentium II and above is recommended).
2. Windows® 98, Windows ME, Windows 2000, or Windows XP.
3. 128 MB RAM.

4. AVR Studio® 4.12 with Service Pack 3 or Atmel Studio.
5. USB port, self-powered (500mA required).
6. Internet Connection for Software download.

Note:
Windows 95 and Windows NT does not support USB, hence cannot be used with AVR Dragon.

3.1.4. Hints

- Always power up the Atmel AVR Dragon first before connecting to or powering up the target
- AVR Dragon needs to sense the target voltage at pin 2 on the SPI(ISP) header or pin 4 on the JTAG header
- This also applies when using the High Voltage interface
- The High Voltage interface is set to 5V. Make sure the target board are running at 5V before using this interface off board.
- VCC header is set to 5V, and can source max. 300mA
- If AVR Dragon is used for programming/debugging targets on the Atmel STK500, the RESET jumper on the STK500 must be removed

3.2. Software and USB Setup

3.2.1. Software and USB Setup

In order to use the Atmel AVR Dragon it is required to install the USB driver. Do not connect the AVR Dragon to the computer before running the USB Setup. USB driver installation is done during the software front-end installation.

Note:
AVR Dragon requires AVR Studio 4.12 with Service Pack 3 or later, or Atmel Studio. Latest version of the Atmel Studio can be found at: http://www.atmel.com/atmelstudio.

Start the Atmel Studio installation. To install the USB driver, check the Install/Upgrade USB Driver checkbox, and the USB Driver will automatically be installed.

3.2.2. Install New Hardware on the Computer

When Atmel Studio and USB driver installation is finished, attach the USB cable to both the PC and the Atmel AVR Dragon. (The AVR Dragon is powered from the USB.) If it is the first time the AVR Dragon is connected to the computer, the box below will appear:

If running Windows XP you need to click "Next" a couple of times. Wait until the installation process completes by itself. It may take from a few seconds up to a few minutes depending on the computer and operating system.

If the USB driver is correctly installed and AVR Dragon is connect to the PC, the green LED next to the USB connector will be lit.

If the software front-end for some reason can't detect the AVR Dragon after the USB setup, try to restart the computer in order to get the driver properly loaded.

3.2.3. Install USB Driver after Atmel Studio is Installed

The USB driver can be installed even after Atmel Studio has been installed by following these steps:

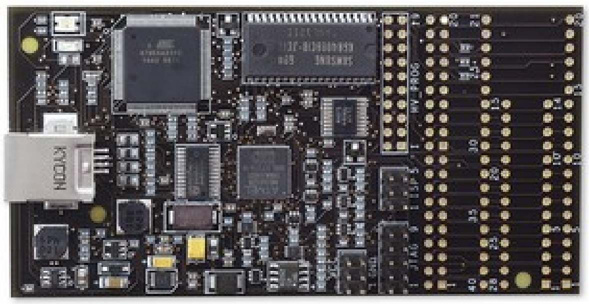
1. Open "**Control Panel**" on the PC (Windows 95 and Windows NT does not support USB).
2. Select "**Add or Remove Programs**".

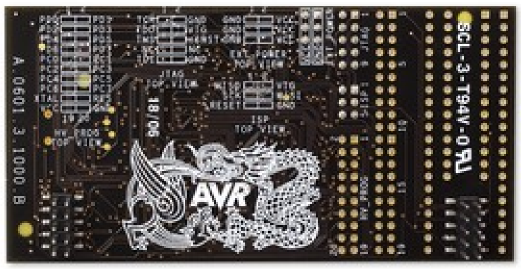
3. Select "**Atmel Studio**" or "**Atmel Studio**" in the list of programs.
4. Click on the "**Change**" button.
5. Select "**Modify**".
6. Select "**Install/upgrade USB Driver**".

The USB driver is now properly installed on the PC.

Note:
The Atmel AVR Dragon requires a USB port that can deliver 500mA (self-powered USB hub).

3.3. Board Description / Headers

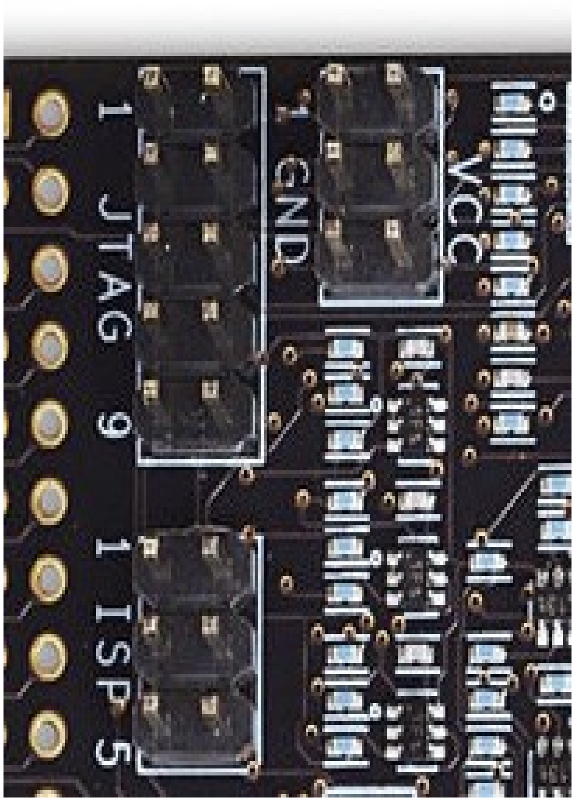




3.3.1. Headers

Out of the box, the Atmel AVR Dragon has the following three header connectors mounted:

- SPI(ISP) Header - Used for SPI(ISP) programming and debugWIRE OCD.
- JTAG Header - Used for JTAG programming and JTAG OCD.
- VCC Header - Used for powering Devices placed in the prototype area, or to power external target boards (max. 300mA). Set to fixed 5V.



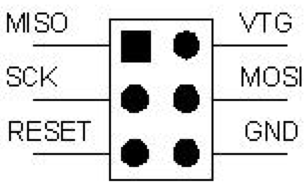
The following headers are not mounted:

- HV_PROG header
- EXPAND header
- 40-pin DIP socket
- 28-pin DIP socket

3.3.1.1. SPI (ISP) Header (mounted)

This 6-pin header uses the standard AVR SPI (ISP) pinout for easy connection to external targets. The signals are level-converted to allow communication with targets running at any voltage between 1.8 and 5.5V.

Note that the target voltage must be applied to pin 2 on the SPI (ISP) header for the Atmel AVR Dragons level converters.



3.3.1.2. JTAG Header (mounted)

The 10-pin JTAG header is a standard pinout JTAG connector. When connecting the Atmel AVR Dragon JTAG header to an external target, the signals are level converted to match the target board voltage. This is done automatically. Note that the AVR Dragon will not power the target through the JTAG interface. The target needs to be powered through a dedicated power supply or by powering it using the VCC connector (5.0V max. 300mA). AVR Dragon needs to sense the target voltage on pin 4 on the JTAG connector.

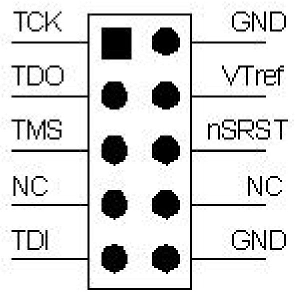
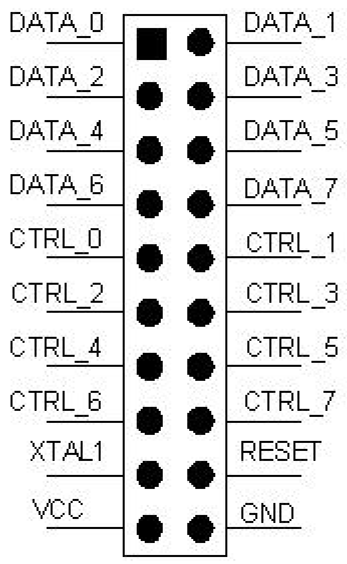


Table 3-1. JTAG Header Pinout and Description

Pin	Signal	I/O	Description
1	TCK	Output	Test Clock, clock signal from AVR Dragon to target JTAG port
2	GND	-	Ground
3	TDO	Input	Test Data Output, data signal from target JTAG port to AVR Dragon
4	VTref	Input	Target reference voltage. VDD from target used to control level-converters.
5	TMS	Output	Test Mode Select, mode select signal from AVR Dragon to target JTAG port
6	nSRST	In/Out-put	Open collector output from adapter to the target system reset. This pin is also an input to the adapter so that the reset initiated on the target may be reported to the AVR Dragon.
7	-	-	Not connected
8	-	-	Not Connected
9	TDI	Output	Test Data Input, data signal from AVR Dragon to target JTAG port
10	GND	-	Ground

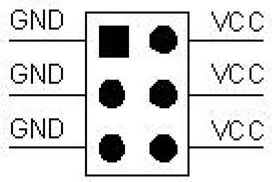
3.3.1.3. HV_PROG Header (not mounted)

The HV_PROG connector contains all signals required to do HVSP or PP programming. The signals on this connector is not level-converted, and should only be connected to the EXPAND connector on the AVR Dragon. You could damage both your target and the Atmel AVR Dragon if you try to do HVSP or PP on an external target board. The signal levels on the HV_PROG header are 5V.



The figure above shows the general pinout of the HV_PROG header. The High Voltage programming pin mapping is not the same for all parts. See the HVSP Description or PP Description chapters for more information on the pinout of this header.

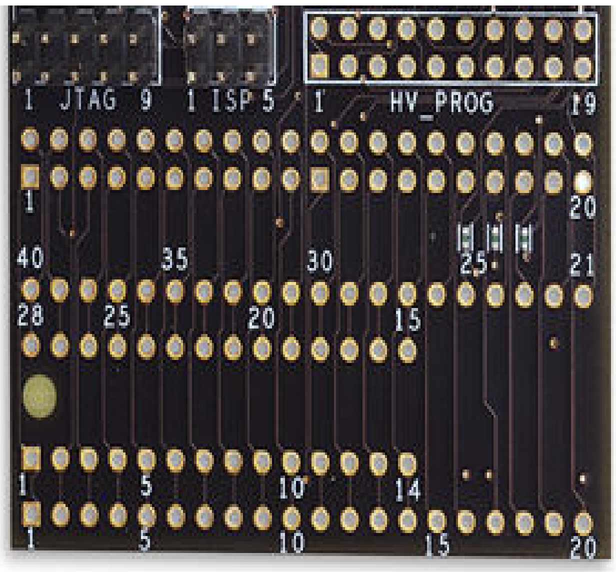
3.3.1.4. VCC Header (mounted)



The VCC Header contains 5.0 Volt VCC and GND that must be used to power the target device placed in the prototype area of the Atmel AVR Dragon board. The voltage can also be used to power an external target board, but it is important that the current consumption is less than 300mA. Note that the AVR Dragon current sourcing capabilities are also limited by the amount of current the Host USB controller can deliver.

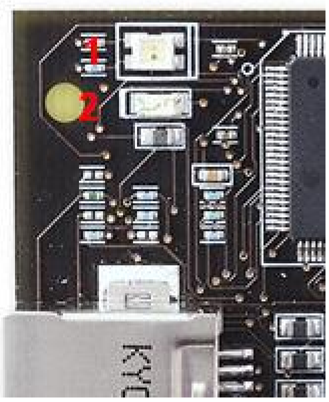
Note:
If the current consumption excedes 300mA, then this may lead to errors or disconnects during programming or debugging. If the AVR Dragon starts misbehaving try to remove the load by applying external power to your circuitry and removing the VCC header connections.

3.3.1.5. EXPAND Header (not mounted)



The expand connector is directly mapped to the 28- and 40-pin DIP sockets. Pin 1 on the connector - is pin one on both the 28 and the 40pin DIP socket. When doing either programming or debugging on-board, the appropriate signals should be routed from the SPI(ISP), JTAG, VCC, and HV_PROG headers to the correct pins on the EXPAND connector. Read the Using the Atmel AVR Dragon Prototype Area section for more information on how to use this function.

3.3.1.6. Status LEDs



Two LEDs show the status of the Atmel AVR Dragon. Check the Troubleshooting Guide to check for solutions if there are any errors.

Table 3-2. LEDs Pinout and Description

LED #	Color	Description
2	Green	Indicates USB traffic
1	Red	Idle, not connected to the software front-end
	Dark	Idle, connected to the software front-end
	Green	Data Transfer
	Yellow	Firmware Upgrade or Initialization

4. Connecting the Atmel AVR Dragon

4.1. Connecting to Target through the JTAG Interface

A minimum of six wires is required to connect Atmel AVR Dragon to the target board. These Signals are TCK, TDO, TDI, TMS, VTref, and GND.

Optional line is the nSRST. The nTRST signal is not used, and is reserved for compatibility with other equipment.

nSRST is used to control and monitor the target reset line. This is however not necessary for correct debugging. But if the application code sets the JTD bit in the MCUCSR, the JTAG Interface will be disabled. For the AVR Dragon to reprogram the target AVR, it will need to have control of the Reset Pin.

The figures in Connecting Atmel AVR Dragon to Target Board shows which JTAG lines should be connected to the target AVR to ensure correct operation. To avoid drive contention on the lines it is recommended that series resistors are placed between the JTAG lines and external circuitry. The value of the resistors should be chosen so that the external circuitry and the AVR device do not exceed their maximum ratings (i.e. sinks or sources to much current).

4.1.1. Connecting Atmel AVR Dragon to Target Board

The JTAG interface consists of a 4-wire Test Access Port (TAP) controller that is compliant with the IEEE® 1149.1 standard. The IEEE standard was developed to provide an industry-standard way to efficiently test circuit board connectivity (Boundary Scan). Atmel AVR devices have extended this functionality to include full Programming and On-Chip Debugging support.

Figure 4-1. Connecting the JTAG connector to external target

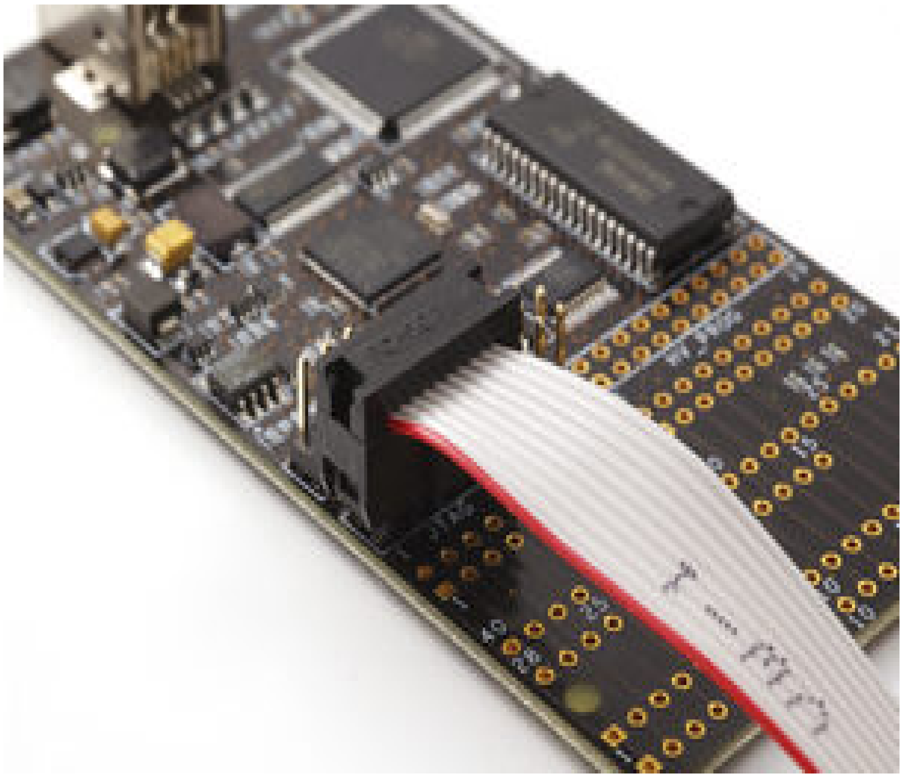


Figure 4-2. Connections needed to access external targets through JTAG interface

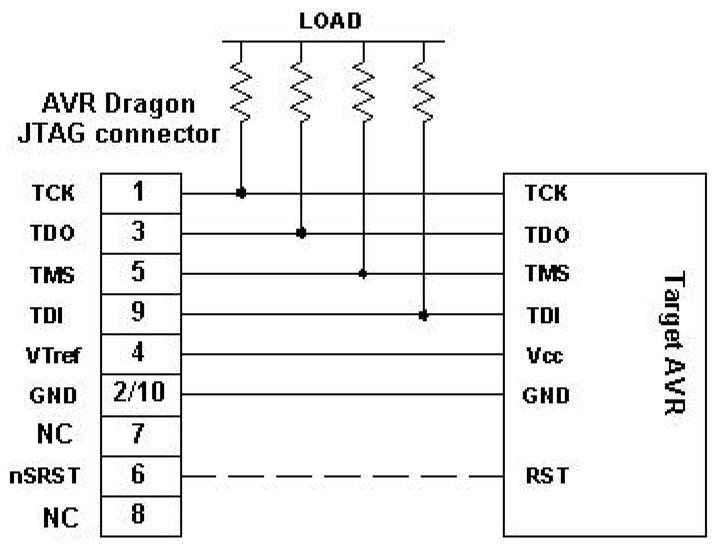


Figure 4-3. JTAG connector pinout

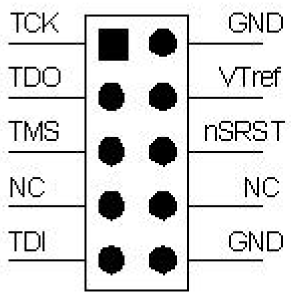


Table 4-1. JTAG Pin Description

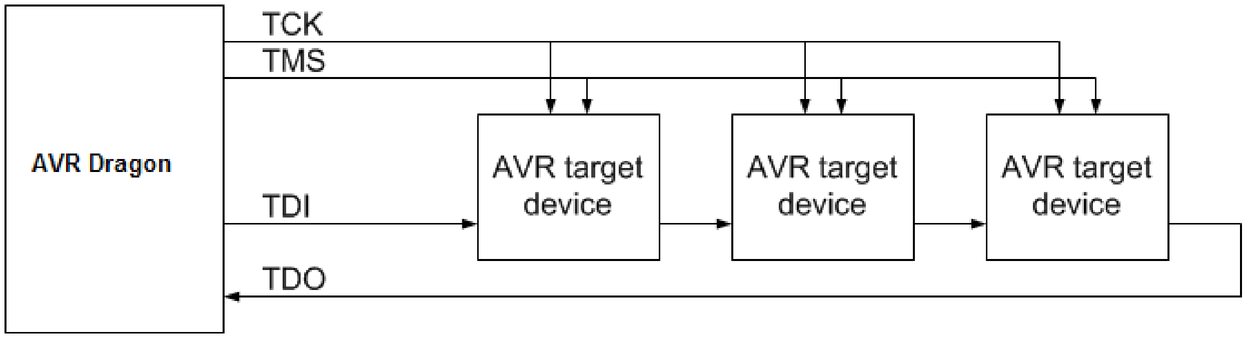
Name	Pin	Description
TCK	1	Test Clock (clock signal from the AVR Dragon into the target device)
TMS	5	Test Mode Select (control signal from the AVR Dragon into the target device)
TDI	9	Test Data In (data transmitted from the AVR Dragon into the target device)
TDO	3	Test Data Out (data transmitted from the target device into the AVR Dragon)
nTRST	8	Test Reset (optional, only on some AVR devices). Used to reset the JTAG TAP controller.
nSRST	6	Reset (optional) Used to reset the target device. Connecting this pin is recommended since it allows the AVR Dragon to hold the target device in a reset state, which can be essential to debugging in certain scenarios.

Name	Pin	Description
VTref	4	Target voltage reference. The AVR Dragon samples the target voltage on this pin in order to power the level converters correctly. The AVR Dragon draws less than 1mA from this pin.
GND	2, 10	Ground. Both must be connected to ensure that the AVR Dragon and the target device share the same ground reference.

4.1.2. Connecting Atmel AVR Dragon to Several Devices Placed in a JTAG Chain

The JTAG interface allows for several devices to be connected to a single interface in a daisy-chain configuration. The target devices must all be powered by the same supply voltage, share a common ground node, and must be connected as shown in Figure 4-4 JTAG Daisy-chain.

Figure 4-4. JTAG Daisy-chain



When connecting devices in a daisy-chain, the following points must be considered:

- All devices must share a common ground, connected to GND on the Atmel AVR Dragon probe
- All devices must be operating on the same target voltage. VTG on the AVR Dragon must be connected to this voltage.
- TMS and TCK are connected in parallel; TDI and TDO are connected in a serial chain.
- nSRST on the AVR Dragon probe must be connected to RESET on the devices if any one of the devices in the chain disables its JTAG port
- "Devices before" refers to the number of JTAG devices that the TDI signal has to pass through in the daisy chain before reaching the target device. Similarly "devices after" is the number of devices that the signal has to pass through after the target device before reaching the AVR Dragon TDO pin.
- "Instruction bits before" and "after" refers to the total sum of all JTAG devices' instruction register lengths which are connected before and after the target device in the daisy chain.
- The total IR length (instruction bits before + AVR IR length + instruction bits after) is limited to a maximum of 256 bits. The number of devices in the chain is limited to 15 before and 15 after.

Daisy chaining example: TDI -> ATmega1280 -> ATxmega128A1 -> ATUC3A0512 -> TDO

In order to connect to the Atmel AVR XMEGA device, the daisy chain settings are:

Devices before: 1

Devices after: 1

Instruction bits before: 4 (AVR 8-bit microcontrollers have 4 IR bits)

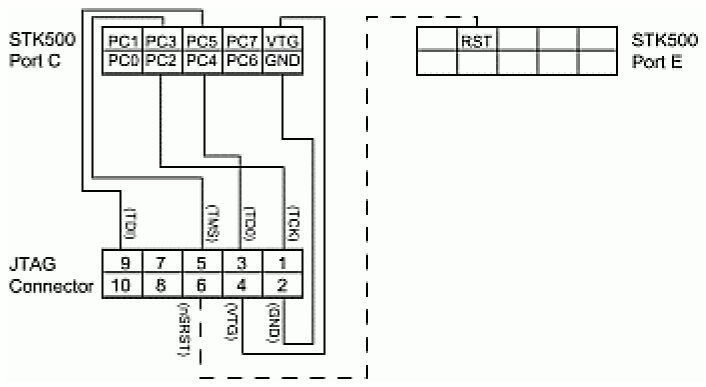
Instruction bits before: 5 (AVR 32-bit microcontrollers have 5 IR bits)

4.1.3. Connecting Atmel AVR Dragon to Atmel STK500

Atmel STK500 does not have a dedicated JTAG interface connector. To connect the Atmel AVR Dragon to the STK500 board, the JTAG Probe must be strapped to the appropriate JTAG Port Pins of the target device using a squid cable. Alternatively an STK500 JTAG adapter can be used, see section The STK500 JTAG adapter. Check the target device datasheet for the location of the JTAG pins on the appropriate device. The figure below shows an example on how the pins should be connected for an ATmega32 on the STK500. Remember to remove the reset jumper on the STK500 if the reset pin is going to be controlled from the AVR Dragon.

Note:
Add-on cards for the STK500 like e.g. STK501/502 may have a dedicated JTAG connector.

4.1.3.1. Example: Connecting Atmel AVR Dragon to Atmel STK500 with ATmega32



4.1.3.2. Atmel STK500 JTAG Adapter

The Atmel STK500 JTAG Adapter, that is shipped with the STK500 (and previously with the JTAGICE mkII), can be used to simplify the connection to the STK500 for Atmel AVR devices with JTAG that mates with socket SCKT3100A3 and SCKT3000D3 on the STK500.

4.1.4. Enabling the JTAG Enable Fuse

If the JTAGEN fuse (JTAG Enable) in the target device is un-programmed, the JTAG Interface will be disabled. This fuse cannot be programmed through the JTAG Interface and must therefore be programmed through the SPI Interface or High Voltage Serial or Parallel Interface.

4.2. Connecting to Target through the debugWIRE Interface

A minimum of three wires are required for communication between Atmel AVR Dragon and the target board with the debugWIRE interface. These signals are RESET, VTref, and GND.

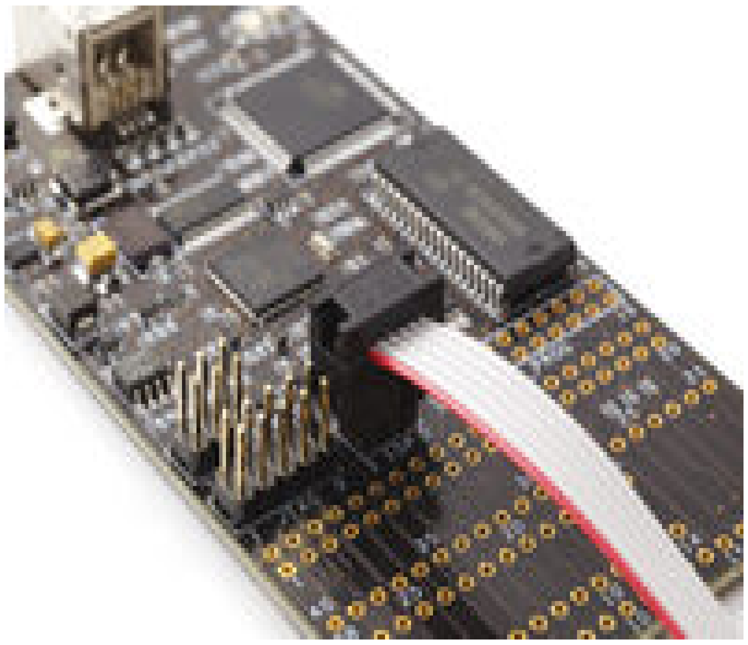
Important!

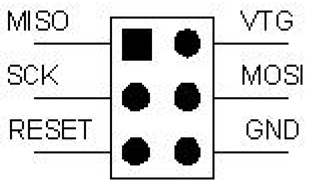
This interface uses only 1 pin, (RESET pin) for communication with the target. To enable the debugWIRE interface on an AVR Device, the debugWIRE Enable fuse (DWEN) must be programmed, (DWEN=0). AVR devices with debugWIRE interface are shipped with the DWEN fuse un-programmed from the factory. The debugWIRE interface itself cannot enable this fuse. The DWEN fuse can be programmed through SPI programming mode, which requires connection to a 6-pin header. For this reason it is recommended to place the full 6-pin SPI connector on your target board to simplify debugging and programming.

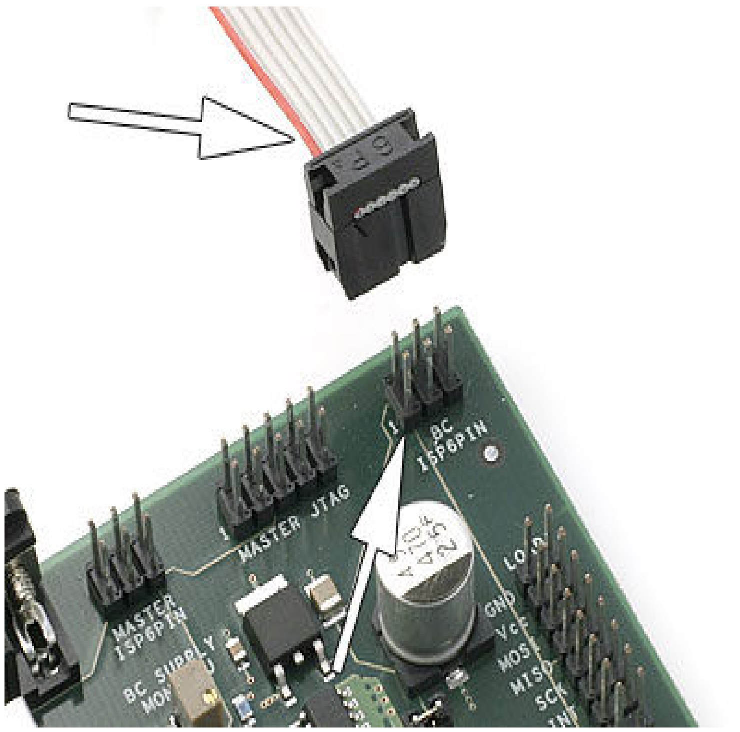
NOTE: When the DWEN fuse is enabled the SPI interface is overridden internally in order for the OCD module to have control over the RESET pin. The debugWIRE OCD is capable of disabling itself temporarily (using the button on the debugging tab in the properties dialog in Atmel Studio), thus releasing control of the RESET line. The SPI interface is then available again (only if the SPIEN fuse is programmed), allowing the DWEN fuse to be un-programmed using the SPI interface. If power is toggled before the DWEN fuse is un-programmed, the debugWIRE module will again take control of the RESET pin. It is HIGHLY ADVISED to simply let Atmel Studio handle setting and clearing of the DWEN fuse!

If using this connection from AVR Dragon on a Atmel STK500, be sure to detach the RESET jumper on the STK500. And connect to the correct ISP header for the actual AVR device, guided by the color code in the STK500 silk-print.

4.2.1. Atmel AVR Dragon debugWIRE Connector







4.2.2. Connecting Atmel AVR Dragon Probe to 6-pins SPI Header using a 6-pin Cable

When the DWEN fuse is programmed, there is only need for the GND, VTref, and RESET lines to be able to use the debugWIRE interface. However, to ease the task of changing between SPI programming mode and debugWIRE mode, it is recommended to use debugWIRE with all six lines connected. The SPI pins will not be driven by the Dragon when running debugWIRE, but pull-up resistors will still be active.

4.2.3. Re-enabling the SPI Interface

By following the description below, the SPI Interface is re-enabled.

1. Connect the Atmel AVR Dragon to the target with SPI (6-pin connection) as described above.
2. Load a project and Start a debug session using the "Start Debugging" command (found under the Debug pull-down menu in Atmel Studio).
3. In the debug menu, you should now be able to choose "Disable debugWIRE and close ".

Note:
Some precautions regarding the RESET line must be taken to ensure proper communication over the debugWIRE interface. If there is a pull-up on the RESET line, this resistor must be larger than 10kΩ, and there should be no capacitive load. The pull-up resistor is not required for debugWIRE functionality. Other logic connected to the RESET line should be removed.

It's not possible to use the debugWIRE interface if the lockbits on the target AVR are programmed. Always be sure that the lockbits are cleared before programming the DWEN fuse and never set the lockbits while the DWEN fuse is programmed. If both the debugWIRE enable fuse (DWEN) and lockbits are set, one can use High Voltage Programming to do a chip erase, hence clear the lockbits. When the lockbits are cleared the debugWIRE interface will be re-enabled.

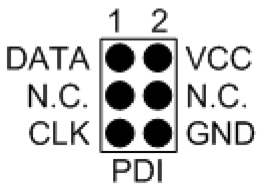
The SPI Interface is only capable of reading fuses, signature, and do a chip erase when the DWEN fuse is unprogrammed.

4.3. PDI Programming

In System Programming using PDI is well suited for programming devices soldered onto external target boards. This section explains how to connect the Atmel AVR Dragon to PDI program an external target. The PDI lines are equipped with level converters that automatically will level shift the AVR Dragon signals to the target board voltage.

It is recommended that a 6-pin header connector with 2.54mm (100 MIL) spacing is placed on the target board to allow easy access to the PDI programming interface. The following pinout should be used.

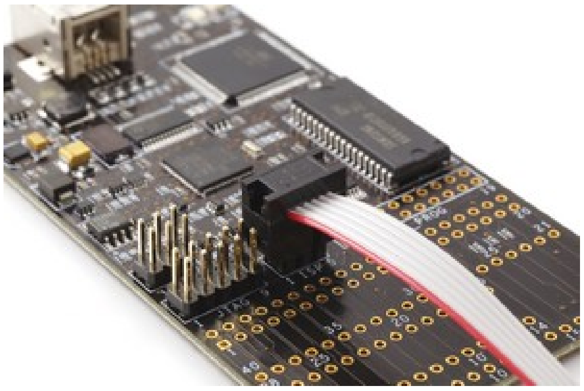
Figure 4-5. 6pin Header Connector with 2.54mm (100 MIL) Spacing

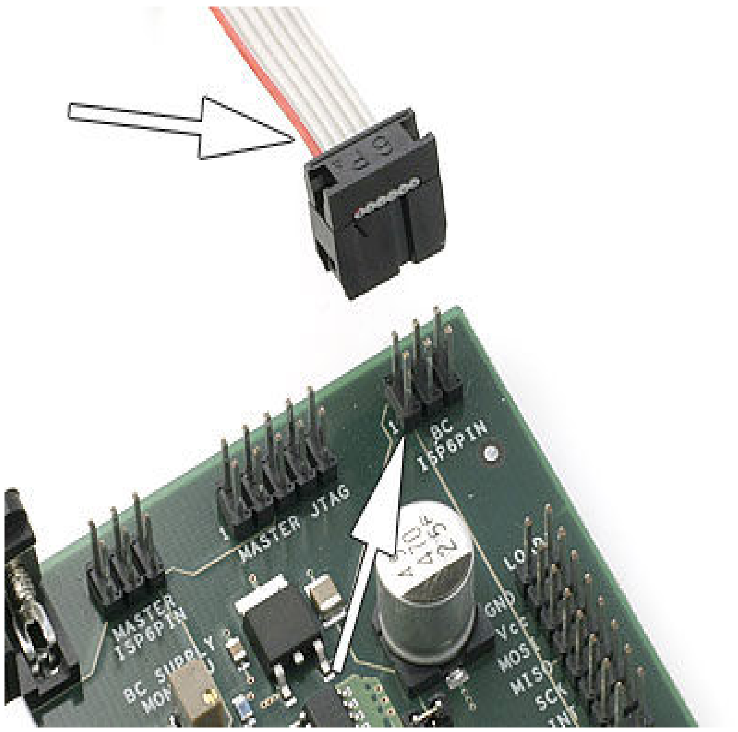


Note:
When connecting the AVR Dragon to the target, connect DATA to DATA pin on the target device, CLK to CLK, and so on.

Note:
AVR Dragon must sense the target voltage on pin 2 on the PDI header in order to set up the level-converters. When using off-board targets there should be no connection between the VCC header and pin 2 of the PDI header.

Connect the 6pin cable from the AVR Dragon to the external target as shown in these pictures:



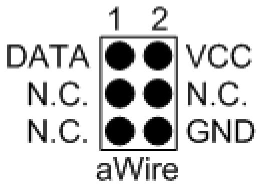


4.4. aWire Programming

The aWire interface makes use of the RESET wire of the Atmel AVR device to allow programming and debugging functions. A special enable sequence is transmitted by the AVR Dragon, which disables the default RESET functionality of the pin.

When designing an application PCB, which includes an AVR with the aWire interface, it is recommended to use the pinout as shown in Figure 4-6 aWire Header Pinout.

Figure 4-6. aWire Header Pinout



Tip:
Since aWire is a half-duplex interface, a pull-up resistor on the RESET line in the order of 47kΩ is recommended to avoid false start-bit detection when changing direction.

The aWire interface can be used as both a programming and debugging interface, all features of the OCD system available through the 10-pin JTAG interface can also be accessed using aWire.

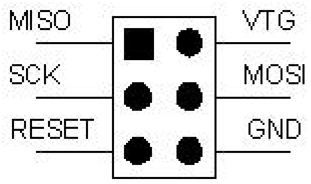
4.5. SPI Programming

In System Programming using SPI is well suited for programming devices soldered onto external target boards. This section explains how to connect the Atmel AVR Dragon to SPI program an external target.

The SPI lines are equipped with level converters that automatically will level shift the AVR Dragon signals to the target board voltage.

It is recommended that a 6-pin header connector with 2.54mm (100 MIL) spacing is placed on the target board to allow easy access to the SPI programming interface. The following pinout should be used.

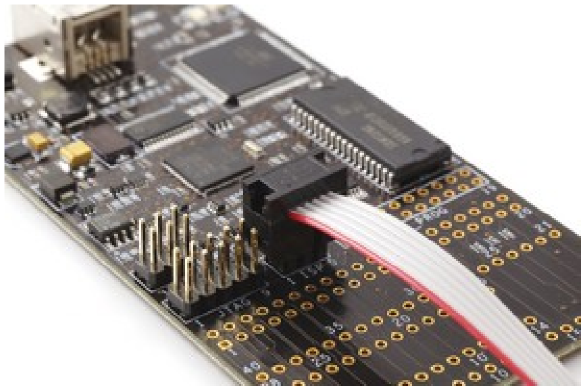
Figure 4-7. 6pin Header Connector with 2.54mm (100 MIL) Spacing

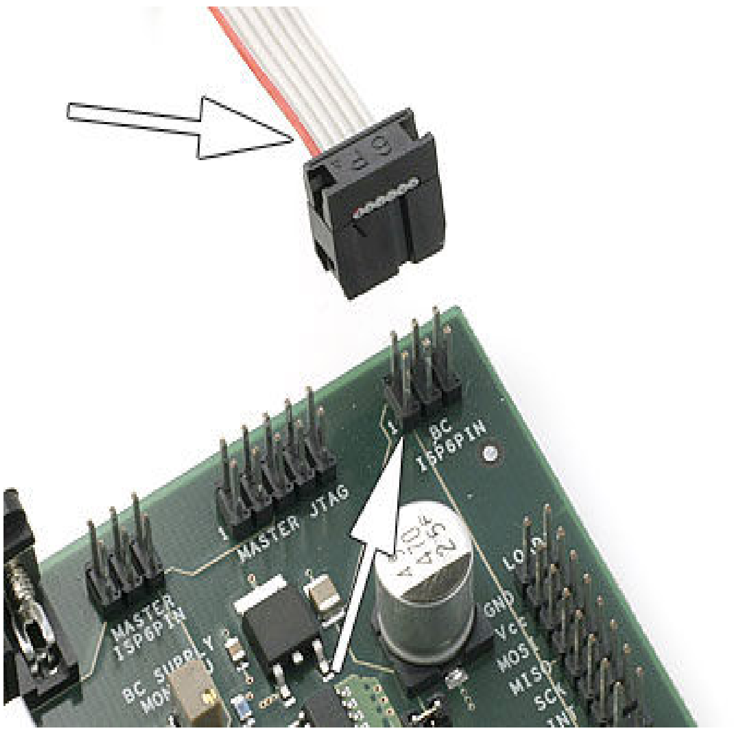


Note:
When connecting the AVR Dragon to the target, connect MISO to MISO pin on the target device, MOSI to MOSI, and so on.

Note:
AVR Dragon must sense the target voltage on pin 2 on the SPI header in order to set up the level-converters. For on-board targets, the voltage must be supplied from pin 2, 4, 6 on the VCC header (5V) into pin 2 (VTG) on the SPI header. When using off-board targets there should be no connection between the VCC header and pin 2 of the SPI header.

Connect the 6pin cable from the AVR Dragon to the external target as shown in these pictures:





debugWIRE OCD interface is also accessed through this SPI header.

Note: The SPI interface is effectively disabled when the debugWIRE enable fuse (DWEN) is programmed, even if SPIEN fuse is also programmed. To re-enable the SPI interface, the 'disable debugWIRE' command must be issued while in a debugWIRE debugging session. Disabling debugWIRE in this manner requires that the SPIEN fuse is already programmed. If Atmel Studio fails to disable debugWIRE, it is probable that the SPIEN fuse is NOT programmed. If this is the case, it is necessary to use a high-voltage programming interface to program the SPIEN fuse. It is HIGHLY ADVISED to simply let Atmel Studio handle setting and clearing of the DWEN fuse!

4.6. Parallel Programming Description

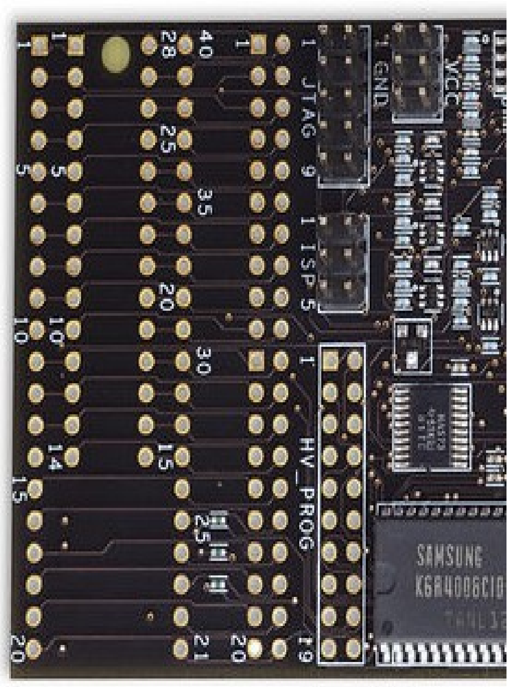
High pin count Atmel AVR devices support the full Parallel Programming (PP) interface. This interface offers high speed programming, and also supports programming all fuse and lock bits in the AVR Device.

Attention:
Extreme care should be taken if using PP mode to program an AVR device on an external target. The PP lines do not have level converters, so it is important that the target board is powered by the AVR Dragon VCC header, and not using its own power supply. In addition the AVR Dragon will apply 12V to the reset pin, so it is important that the target board is designed to handle 12V on this line.

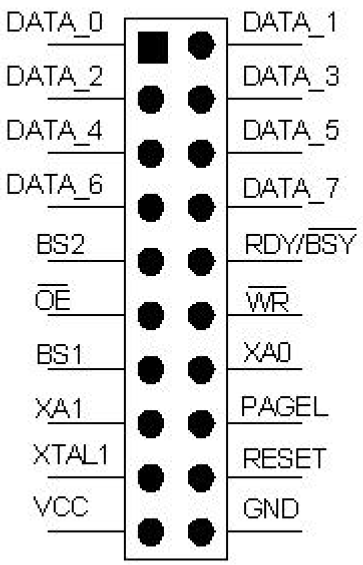
Note:
The target voltage, i.e. the 5V from the VCC header must be applied to either pin 2 on the SPI header or pin 4 on the JTAG header. This is because the AVR Dragon must read the target voltage.

To avoid damaging the Target Board, the AVR Dragon or both, it is recommended to **only** use PP mode on devices placed in the 28/40 pin DIP socket on the AVR Prototype area on the AVR Dragon.

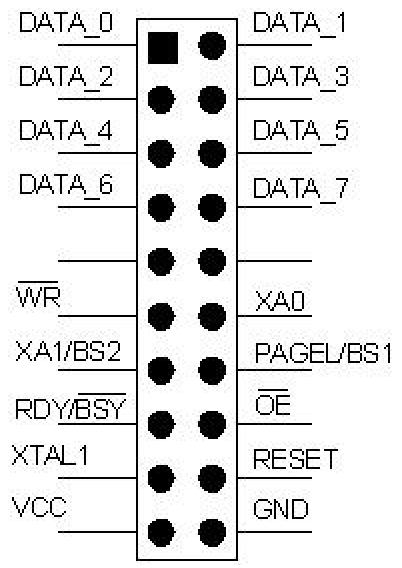
Figure 4-8. Prototype Area



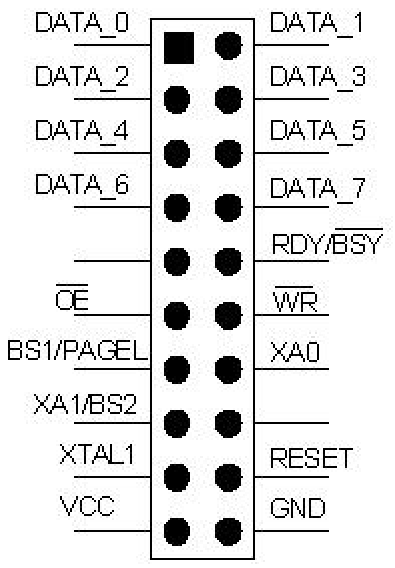
The HV_PROG header pinout is listed below. This is the standard pinout for about all Atmel AVR parts. However, the pinout on the HV_PROG header is slightly different for some parts. These exceptions are listed further down in this page.



For ATtiny26/261/461/861 the HV_PROG header will have this pinout:



For ATtiny2313 the HV_PROG header will have this pinout:



See the Device Connection Sheet section for information on how to connect AVR Dragon for PP programming.

4.7. High Voltage Serial Programming Description

Low pin count Atmel AVR devices do not have enough I/O pins to support the full Parallel Programming interface. These devices use HVSP programming instead, which is a serial version of the Parallel Programming interface.