Programmers and Debuggers

JTAGICE3

USER GUIDE

Introduction

The Atmel JTAGICE3 is a powerful development tool for debugging and programming ARM® Cortex®-M based Atmel® SAM and Atmel AVR® microcrontrollers with On-Chip Debug capability.

- Programming and on-chip debugging of all Atmel AVR 32-bit Microcontrollers on both JTAG and aWire interfaces
- Programming and on-chip debugging of all Atmel AVR XMEGA® 8-bit Microcontroller Family devices on both JTAG and PDI 2-wire interfaces
- Programming (JTAG, SPI, and UPDI) and debugging of all Atmel AVR 8-bit Microcontrollers with OCD support on JTAG, debugWIRE, or UPDI interfaces
- Programming and debugging of all Atmel SAM ARM Cortex-M based microcontrollers on both SWD and JTAG interfaces (firmware version 3.0 and later)

Table of Contents

1. Features

1.1. Atmel JTAGICE3 Features

- Fully compatible with Atmel Studio
- Supports programming and debugging of all Atmel AVR UC3 32-bit microcontrollers
- Supports programming and debugging of all 8-bit AVR XMEGA devices
- Supports programming and debugging of all 8-bit Atmel megaAVR and tinyAVR devices with OCD
- Supports programming and debugging of all SAM ARM Cortex-M based microcontrollers
- Target operating voltage range of 1.65V to 5.5V
- Draws less than 3mA from target VTref during operation
- Supports JTAG clock frequencies from 32kHz to 15MHz
- Supports PDI clock frequencies from 32kHz to 10MHz
- Supports debugWIRE baud rates from 4kbit/s to 0.5Mbit/s
- Supports aWire baud rates from 7.5kbit/s to 7.5Mbit/s
- Supports SPI clock frequencies from 8kHz to 1.875MHz
- Supports UPDI baud rates from up to 750kbit/s
- Supports SWD clock frequencies from 32kHz to 2MHz
- USB 2.0 high-speed host interface
- ITM serial trace capture at up to 1MB/s
- Supports 10-pin 50-mil JTAG connector, as well as 10-pin 100mil JTAG, 6-pin 50-mil SPI, and 6-pin 100-mil SPI interfaces using adapters

1.2. System Requirements

The Atmel JTAGICE3 unit requires that a front-end debugging environment as Atmel Studio or AVR Studio® 5.0 or later is installed on your computer.

The JTAGICE3 should be connected to the host computer using the USB cable provided.

2. Getting Started with the Atmel JTAGICE3

2.1. Kit Contents

The Atmel JTAGICE3 kit contains these items:

- JTAGICE3 unit
- USB cable (1.8m, high-speed, Mini-B)
- Adapter board containing 50-mil SPI, 100-mil SPI, and 100-mil JTAG adapters
- 50-mil JTAG cable
- 50-mil 10-pin mini squid cable
- AVR Technical Library DVD

Figure 2-1. JTAGICE3 Kit Contents

2.2. Assembling the Atmel JTAGICE3

The Atmel JTAGICE3 unit is shipped with no cables attached. Two cable options are provided in the accessory kit:

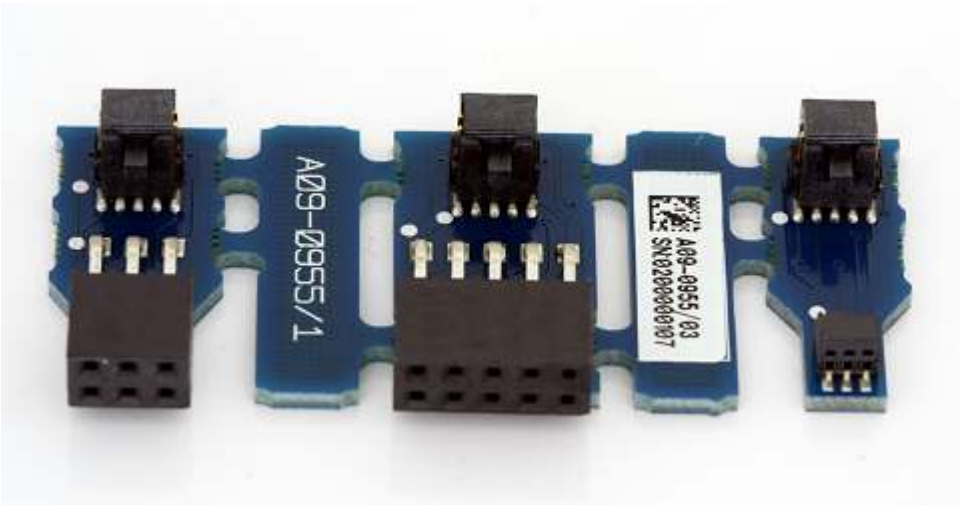
- 50-mil 10-pin IDC flat cable
- 50-mil 10-pin mini-squid cable with 10 x 100-mil sockets

Figure 2-2. JTAGICE3 Cables

For most purposes, the 50-mil 10-pin IDC flat cable can be used along with one of the adapters provided in the accessory kit. Three adapters are provided in one small PCB panel section. To separate the adapters, simply break off each one by gently bending the panel backwards and forwards. Take care to avoid any sharp edges, which may result from the separation process. The following adapters are included:

- A09-0955 - A 100-mil 10-pin JTAG adapter
- A09-0955 - B 100-mil 6-pin SPI/debugWIRE/PDI/aWire adapter
- A09-0955 - C 50-mil 6-pin SPI/debugWIRE/PDI/aWire adapter

Figure 2-3. JTAGICE3 Adapters



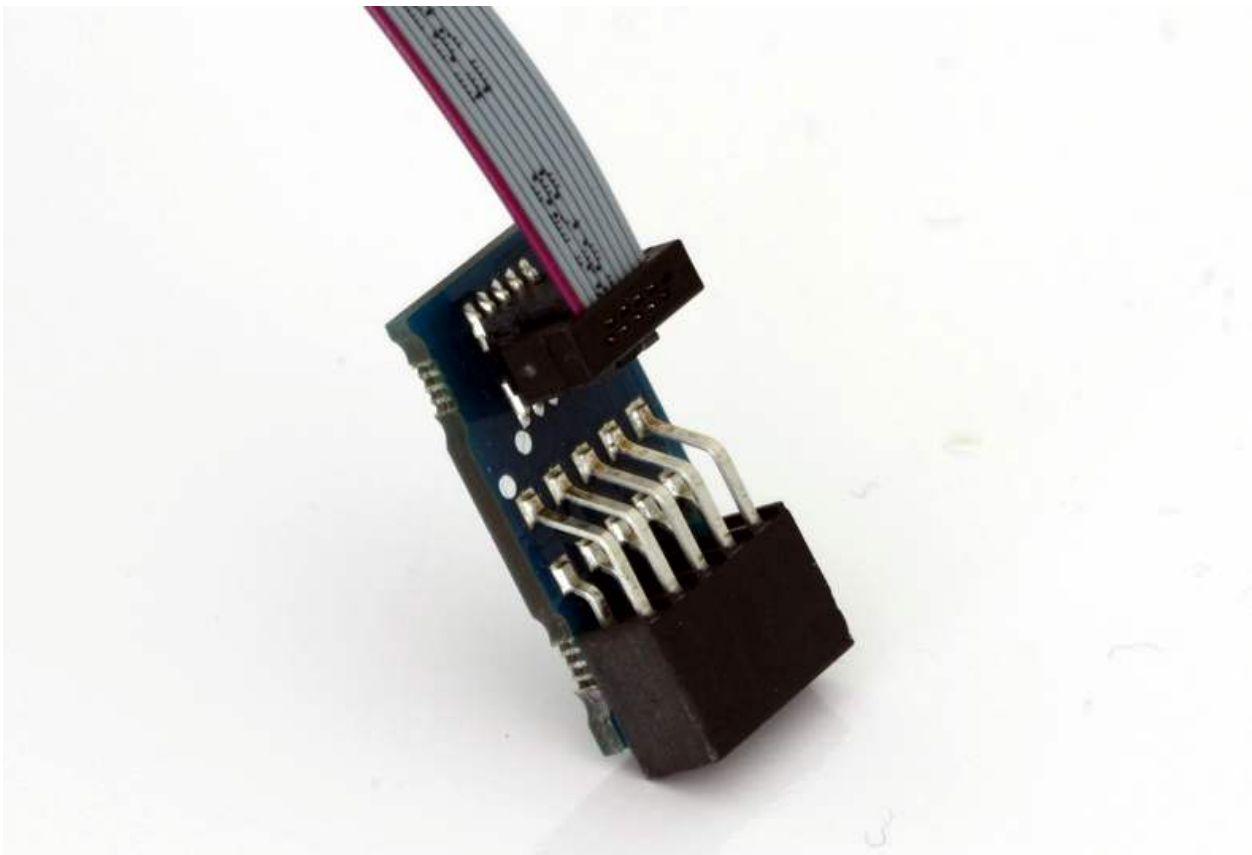
Note that a 50-mil JTAG adapter has not been provided - this is because the 50-mil 10-pin IDC cable can be used to connect directly to a 50-mil JTAG header. For the part number of the component used for the 50-mil 10-pin connector, see target connectors part numbers.

To assemble your JTAGICE3 into its 'default' configuration, connect the 10-pin 50-mil IDC cable to the unit as shown below. Be sure to orient the cable so that the red wire (pin 1) on the cable aligns with the triangular indicator on the blue belt of the enclosure. The cable should connect upwards from the unit.

Figure 2-4. Atmel JTAGICE3 Cable Connection

Next, snap off the 100-mil 10-pin JTAG adapter from the adapter panel, and connect it to the other side of the IDC cable. Your Atmel JTAGICE3 is now ready to use in its basic JTAG configuration.

Figure 2-5. JTAGICE3 Probe Adapter Connection



2.3. Opening the Atmel JTAGICE3

Note: For normal operation, the Atmel JTAGICE3 unit must not be opened. Opening the unit is done at your own risk. Anti-static precautions should be taken.

The JTAGICE3 enclosure consists of three separate plastic components - top cover, bottom cover, and blue belt - which are snapped together during assembly. To open the unit, simply insert a large flat screwdriver into the openings in the blue belt, apply some inward pressure, and twist gently. Repeat the process on the other snapper holes, and the top cover will pop off.

Figure 2-6. Opening the JTAGICE3 (1)

Figure 2-7. Opening the JTAGICE3 (2)

Figure 2-8. Opening the JTAGICE3 (3)

To close the unit again, simply align the top and bottom covers correctly, and press together firmly.

2.4. Powering the Atmel JTAGICE3

The Atmel JTAGICE3 is powered by the USB bus voltage. It requires less than 100mA to operate, and can therefore be powered through a USB hub. The power LED will illuminate when the unit is plugged in. When not connected in an active programming or debugging session, the unit will enter low-power consumption mode to preserve your computer's battery. The JTAGICE3 cannot be powered down - it should be unplugged when not in use.

2.5. Connecting to the Host Computer

Before plugging in the Atmel JTAGICE3 for the first time, be sure to install the USB driver on the host computer. This is done automatically when installing the front-end software provided free by Atmel. See www.atmel.com for further information or to download the latest front-end software.

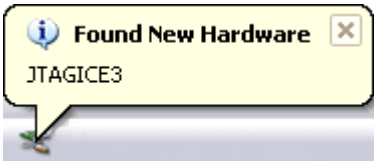
The JTAGICE3 must be connected to an available USB port on the host computer using the USB cable provided. The JTAGICE3 contains a USB 2.0 compliant controller, and can operate in both full-speed and high-speed modes. For best results, connect the JTAGICE3 directly to a USB 2.0 compliant high-speed hub on the host computer using the cable provided.

2.6. USB Driver Installation

2.6.1. Windows

When installing the Atmel JTAGICE3 on a computer running Microsoft® Windows®, the USB driver is loaded when the JTAGICE3 is first plugged in.

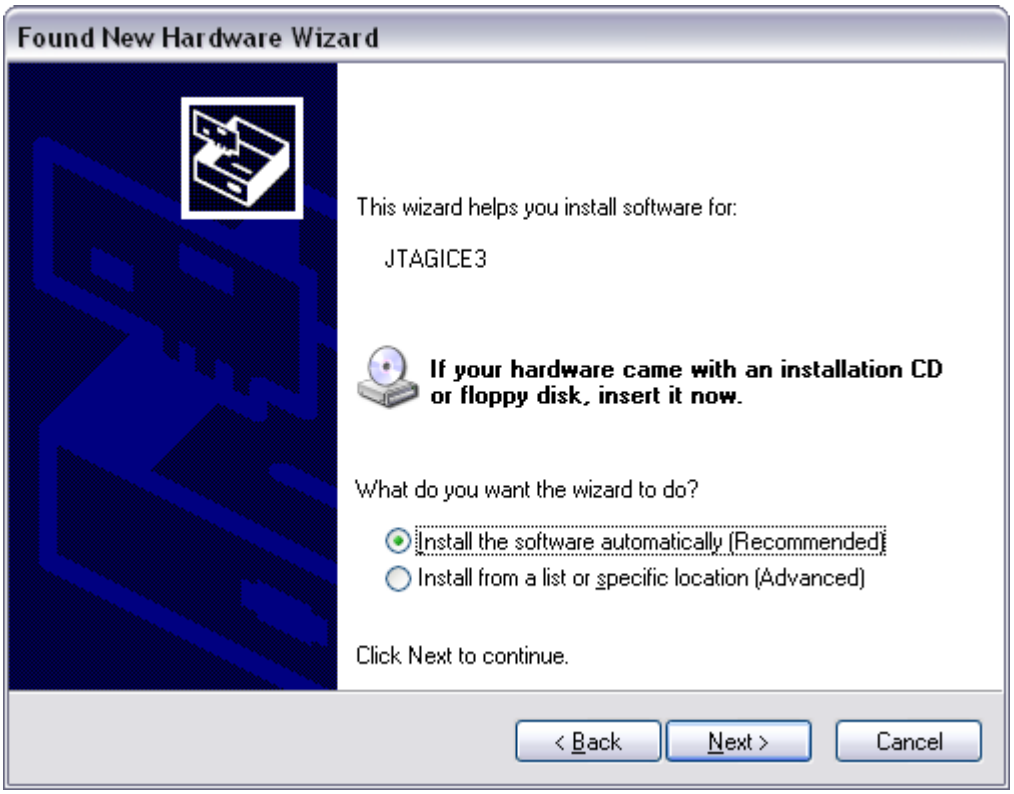
Note: Be sure to install the front-end software packages before plugging the unit in for the first time!



Proceed with the default ("recommended") options through the New Hardware Wizard.

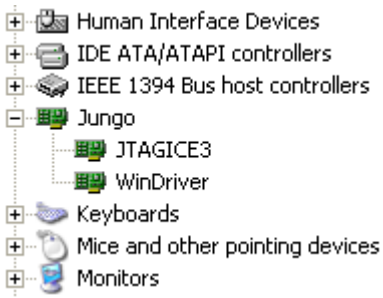
Figure 2-9. Installing the JTAGICE3 USB Driver

Figure 2-10. Installing the JTAGICE3 USB Driver

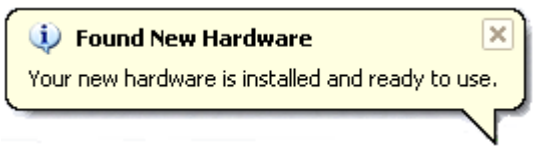


If it is not detected automatically, point the wizard to the device driver (provided by Jungo) called jtagice3.inf which is stored in the <windows_root>\inf folder.

Once successfully installed, the JTAGICE3 will appear in the device manager as a "Jungo" device.



Your JTAGICE3 is now ready to use.



Note: Firmware version 3 and later uses a HID interface for communication. The USB driver provided by the operating system is loaded automatically in this case.

3. Connecting the Atmel JTAGICE3

3.1. Connecting to a JTAG Target

The Atmel JTAGICE3 probe has a 50-mil 10-pin JTAG connector accessible on the front of the tool's enclosure. The kit includes a 50-mil 10-pin cable, which can be used to connect directly to a 50-mil JTAG header on your target board. Should your target board be fitted with a 100-mil JTAG header (e.g.: Atmel STK®600), the 100-mil adapter provided (A08-0735-A) in the kit can be used. If your target board does not have a compliant 10-pin JTAG header in 50- or 100-mil, you can map to a custom pinout using the 10-pin "mini-squid" cable provided, which gives access to 10 individual 100-mil sockets.

3.1.1. Using the JTAG 10-pin Connector

The pinout for the 10-pin JTAG connector is shown in Figure 4-2.

Be sure to use the correct orientation of the 10-pin header when connecting the JTAGICE3 to the target application PCB. The red wire on the probe cable indicates PIN1, as does the 'dot' on the 100-mil adapter.

3.2. Connecting to an aWire Target

The aWire interface requires only one data line in addition to V_{CC} and GND. The 10-pin mini-squid cable should be used to connect between the Atmel JTAGICE3 target connector and the target board. Three connections are required, as described in the table below.

Table 3-1. Connecting to aWire using the Mini-squid Cable

JTAGICE3 pins	Target pins	Mini-squid pin	aWire pinout
Pin 1 (TCK)		1	
Pin 2 (GND)	GND	2	6
Pin 3 (TDO)	DATA	3	1
Pin 4 (VTG)	VTG	4	2
Pin 5 (TMS)		5	
Pin 6 (nSRST)		6	
Pin 7 (Not connected)		7	
Pin 8 (nTRST)		8	

JTAGICE3 pins	Target pins	Mini-squid pin	aWire pinout
Pin 9 (TDI)		9	
Pin 10 (GND)		0	

3.3. Connecting to a PDI Target

The pinout for the 6-pin PDI connector is shown in Figure 4-5.

The adapters provided in the kit (A08-0735-B and A08-0735-C) can be used to connect to this pinout in either 50-mil or 100-mil options.

Be sure to use the correct orientation of the 6-pin header when connecting the Atmel JTAGICE3 to the target application PCB. A 'dot' on the adapter board indicates pin 1.

When connecting to a target that does not have the standard 6-pin header, you can use the 10-pin mini-squid cable between the JTAGICE3 target connector and the target board. Four connections are required, and the table below describes where to connect them.

Note: There is a difference from the JTAGICE mkII JTAG probe, where PDI_DATA is connected to pin 9. The JTAGICE3 is compatible with the pinout used by the Atmel AVR ONE! and Atmel AVR Dragon™ products.

Table 3-2. Connecting to PDI using the Mini-squid Cable

JTAGICE3 pin	Target pins	Mini-squid pin	Atmel STK600 PDI pinout
Pin 1 (TCK)		1	
Pin 2 (GND)	GND	2	6
Pin 3 (TDO)	PDI_DATA	3	1
Pin 4 (VTG)	VTG	4	2
Pin 5 (TMS)		5	
Pin 6 (nSRST)	PDI_CLK	6	5
Pin 7 (Not connected)		7	
Pin 8 (nTRST)		8	
Pin 9 (TDI)		9	
Pin 10 (GND)		0	

3.4. Connecting to a debugWIRE Target

The pinout for the 6-pin debugWIRE (SPI) connector is shown in Figure 4-7.

The adapters provided in the kit (A08-0735-B and A08-0735-C) can be used to connect to this pinout in either 50-mil or 100-mil options.

Be sure to use the correct orientation of the 6-pin header when connecting the Atmel JTAGICE3 to the target application PCB. A 'dot' on the adapter board indicates pin 1.

Although the debugWIRE interface only requires one signal line (RESET), V_{CC}, and GND to operate correctly, it is advised to have access to the full SPI connector so that the debugWIRE interface can be enabled and disabled using SPI programming.

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!

It is not possible to use the debugWIRE Interface if the lockbits on the target Atmel 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, and thus clear the lockbits. When the lockbits are cleared the debugWIRE Interface will be re-enabled. The SPI Interface is only capable of reading fuses, reading signature, and performing a chip erase when the DWEN fuse is un-programmed.

Table 3-3. Connecting to debugWIRE using the Mini-squid Cable

JTAGICE3 pin	Target pins	Mini-squid pin
Pin 1 (TCK)		1
Pin 2 (GND)	GND	2
Pin 3 (TDO)		3
Pin 4 (VTG)	VTG	4
Pin 5 (TMS)		5
Pin 6 (nSRST)	RESET	6
Pin 7 (Not connected)		7
Pin 8 (nTRST)		8

JTAGICE3 pin	Target pins	Mini-squid pin
Pin 9 (TDI)		9
Pin 10 (GND)		0

3.5. Connecting to an SPI Target

The pinout for the 6-pin SPI connector is shown in Figure 4-8.

The adapters provided in the kit (A08-0735-B and A08-0735-C) can be used to connect to this pinout in either 50-mil or 100-mil options.

Be sure to use the correct orientation of the 6-pin header when connecting the Atmel JTAGICE3 to the target application PCB. A 'dot' on the adapter board indicates pin 1.

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!

The 10-pin mini-squid cable can also be used to connect between the Atmel JTAGICE3 target connector and the SPI target board. Six connections are required, as described in the table below.

Table 3-4. Connecting to SPI using the Mini-squid Cable

JTAGICE3 pins	Target pins	Mini-squid pin	SPI pinout
Pin 1 (TCK)	SCK	1	3
Pin 2 (GND)	GND	2	6
Pin 3 (TDO)	MISO	3	1
Pin 4 (VTG)	VTG	4	2
Pin 5 (TMS)		5	
Pin 6 (nSRST)	/RESET	6	5
Pin 7 (Not connected)		7	
Pin 8 (nTRST)		8	
Pin 9 (TDI)	MOSI	9	4
Pin 10 (GND)		0	

3.6. Connecting to an SWD Target

The ARM SWD interface is s subset of the JTAG interface, making use of TCK and TMS pins, which means that when connecting to an SWD device, the 10-pin JTAG connector can technically be used. The ARM JTAG and AVR JTAG connectors are however not pin-compatible, so this depends upon the layout of the target board in use. When using STK600 or a board making use of the AVR JTAG pinout, the standard JTAG connector can be used. When connecting to a board which makes use of the ARM JTAG pinout, the 10-pin "mini-squid" cable must be used. (There are no adapters available from Atmel to make this mapping.)

Table 3-5. Connecting to SWD using the Mini-squid Cable

JTAGICE3 pin	Target pins	Mini-squid pin	ARM SWD pin
Pin 1 (TCK)	SWDCLK	1	4 (SWDCLK)
Pin 2 (GND)	GND	2	3, 5, 9 (GND)
Pin 3 (TDO)	SWO (optional)	3	6
Pin 4 (VTG)	VTG	4	1 (VCC)
Pin 5 (TMS)	SWDIO	5	2 (SWDIO)
Pin 6 (nSRST)	RESET	6	10 (nRESET)
Pin 7 (Not connected)		7	
Pin 8 (nTRST)		8	
Pin 9 (TDI)		9	
Pin 10 (GND)		0	3, 5, 9 (GND)

3.7. Connecting to a UPDI Target

The pinout for the 6-pin PDI connector is shown in UPDI Physical Interface.

However, the JTAGICE3 has hardware, which is not able to drive the 1-wire UPDI interface on pin 3 (AVR header) without driving pin 6 (AVR header) at the same time. For this reason it is recommended to use the 10-pin mini-squid cable for UPDI.

Note: Atmel-ICE and newer tools are capable of direct connection to the UPDI header.

Table 3-6. Connecting to UPDI using the Mini-squid Cable

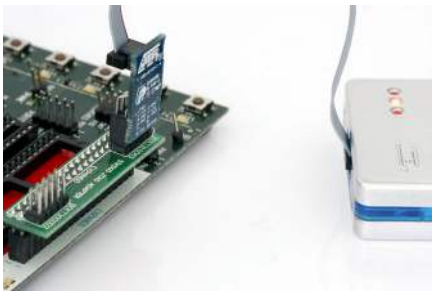
JTAGICE3 pin	Target pins	Mini-squid pin	Atmel STK600 UPDI pinout
Pin 1 (TCK)		1	
Pin 2 (GND)	GND	2	6
Pin 3 (TDO)	UPDI_DATA	3	1
Pin 4 (VTG)	VTG	4	2
Pin 5 (TMS)		5	
Pin 6 (nSRST)		6	
Pin 7 (not connected)		7	
Pin 8 (nTRST)		8	
Pin 9 (TDI)		9	
Pin 10 (GND)		0	

3.8. Using the Atmel JTAGICE3 with Atmel STK500

The Atmel STK500 starter kit can be used to house Atmel AVR devices to which the Atmel JTAGICE3 can connect through JTAG, debugWIRE, and SPI interfaces.

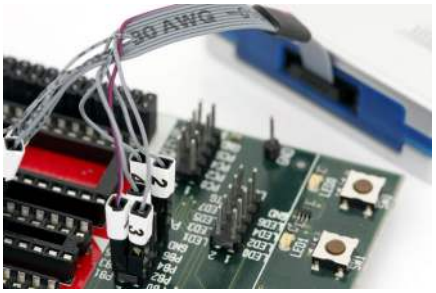
When connecting to a JTAG target, simply use the ATSTK500_JTAG_ADAPTER.

If you do not have an STK500 JTAG adapter available, the 10-pin "squid" cable can also be used to connect directly to the device's JTAG port on PORTC[5::2] of the STK500.



Connecting to debugWIRE and SPI targets is done using the 6-pin 100 mil adapter. When using the debugWIRE interface, be sure to remove the STK500's RESET jumper to allow the reset line to be driven as required.

Alternatively, the JTAGICE3 can be connected to any target interface using the 10-pin mini-squid cable provided.



3.9. Using the Atmel JTAGICE3 with Atmel STK600

The Atmel STK600 starter kit can be used to house AVR devices to which the Atmel JTAGICE3 can connect through the JTAG, debugWIRE, PDI, SPI, and aWire interfaces.

When connecting to a JTAG target, simply use the 10-pin 100mil adapter provided to connect to the JTAG connector on the STK600.

When connecting to a PDI, debugWIRE, SPI, or aWire target, simply use the 6-pin 100mil adapter provided to connect to the SPI/PDI header connector.

4. On-chip Debugging

4.1. Introduction to On-chip Debugging (OCD)

A traditional *Emulator* is a tool which tries to imitate the exact behavior of a target device. The closer this behavior is to the actual device's behavior, the better the emulation will be.

The Atmel JTAGICE3 is not a traditional *Emulator*. Instead, the JTAGICE3 interfaces with the internal On-Chip Debug system inside the target Atmel AVR device, providing a mechanism for monitoring and controlling its execution. In this way the application being debugged is not *emulated*, but actually executed on the real AVR target device.

With an OCD system, the application can be executed whilst maintaining exact electrical and timing characteristics in the target system – something not technically realizable with a traditional *emulator*.

Run Mode

When in Run mode, the execution of code is completely independent of the JTAGICE3. The JTAGICE3 will continuously monitor the target AVR to see if a break condition has occurred. When this happens the OCD system will interrogate the device through its debug interface, allowing the user to view the internal state of the device.

Stopped Mode

When a breakpoint is reached, program execution is halted, but all I/O will continue to run as if no breakpoint had occurred. For example, assume that a USART transmit has just been initiated when a breakpoint is reached. In this case the USART continues to run at full speed completing the transmission, even though the core is in stopped mode.

Hardware Breakpoints

The AVR OCD module contains a number of program counter comparators implemented in hardware. When the program counter matches the value stored in one of the comparator registers, the OCD enters stopped mode. Since hardware breakpoints require dedicated hardware on the OCD module, the number of breakpoints available depends upon the size of the OCD module implemented on the AVR target. Usually one such hardware comparator is 'reserved' by the debugger for internal use. For more information on the hardware breakpoints available in the various OCD modules, see the OCD implementations section.

Software Breakpoints

A software breakpoint is a BREAK instruction placed in program memory on the target device. When this instruction is loaded, program execution will break and the OCD enters stopped mode. To continue execution a "start" command has to be given from the OCD. Not all AVR devices have OCD modules supporting the BREAK instruction. For more information on the software breakpoints available in the various OCD modules, see the OCD implementations section.

4.2. Physical Interfaces

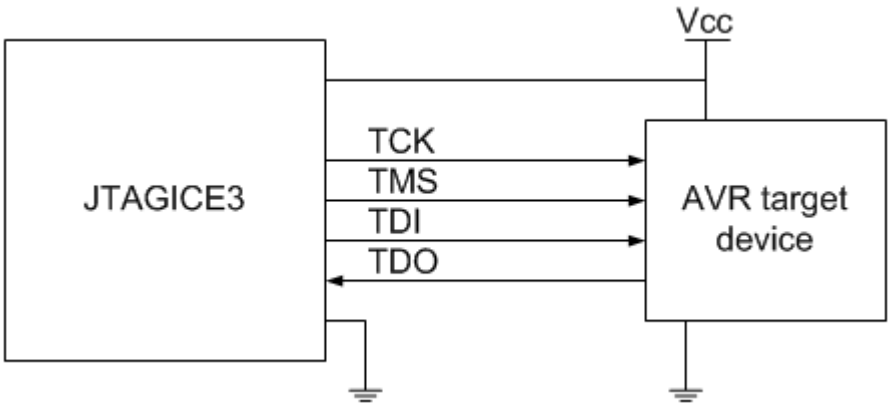
The Atmel JTAGICE3 supports several hardware interfaces as described in the sections that follow.

4.2.1. JTAG

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. JTAG Interface Basics



When designing an application PCB, which includes an Atmel AVR with the JTAG interface, it is recommended to use the pinout as shown in Figure 4-2. The JTAGICE3 can connect to both 100-mil and 50-mil variants of this pinout.

Figure 4-2. JTAG Header Pinout

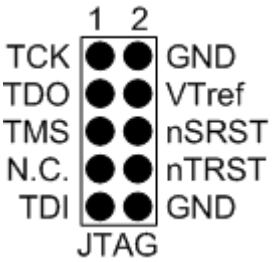


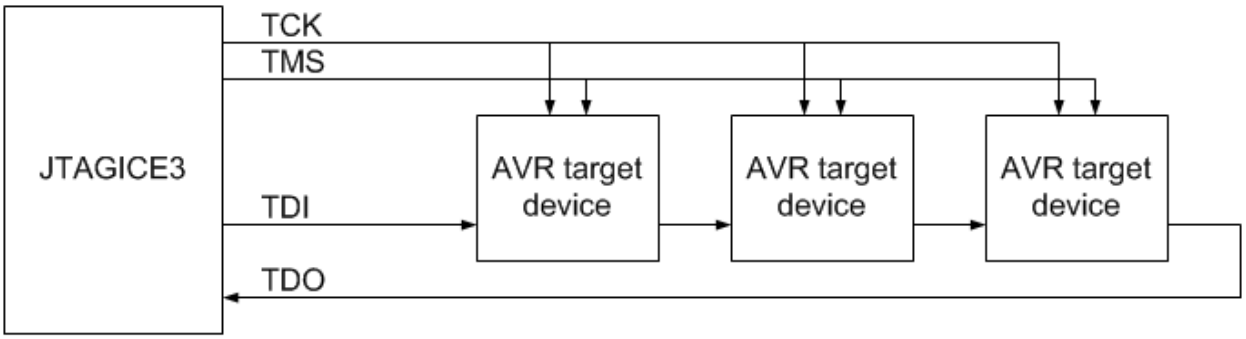
Table 4-1. JTAG Pin Description

Name	Pin	Description
TCK	1	Test Clock (clock signal from the JTAGICE3 into the target device)
TMS	5	Test Mode Select (control signal from the JTAGICE3 into the target device)
TDI	9	Test Data In (data transmitted from the JTAGICE3 into the target device)
TDO	3	Test Data Out (data transmitted from the target device into the JTAGICE3)
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 JTAGICE3 to hold the target device in a reset state, which can be essential to debugging in certain scenarios.
VTG	4	Target voltage reference. The JTAGICE3 samples the target voltage on this pin in order to power the level converters correctly. The JTAGICE3 draws less than 1mA from this pin.
GND	2, 10	Ground. Both must be connected to ensure that the JTAGICE3 and the target device share the same ground reference.

Tip: Remember to include a decoupling capacitor between pin 4 and GND.

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 be connected as shown in the figure below.

Figure 4-3. 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 JTAGICE3 probe
- All devices must be operating on the same target voltage. VTG on the JTAGICE3 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 JTAGICE3 probe must be connected to RESET on the devices if any 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 JTAGICE3 TDO pin.
- "Instruction bits before" and "after" refers to the 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 + Atmel 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 (8-bit AVR devices have four IR bits)

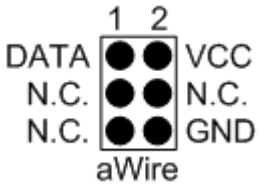
Instruction bits before: 5 (32-bit AVR devices have five IR bits)

4.2.2. aWire

The aWire interface makes use of the RESET wire of the AVR device to allow programming and debugging functions. A special enable sequence is transmitted by the Atmel JTAGICE3, which disables the default RESET functionality of the pin.

When designing an application PCB, which includes an Atmel AVR with the aWire interface, it is recommended to use the pinout as shown in the figure below. The JTAGICE3 ships with both 100-mil and 50-mil adapters supporting this pinout.

Figure 4-4. 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.

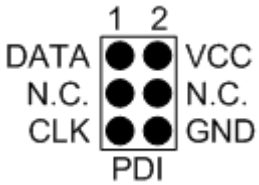
The aWire interface can be used as both a programming and debugging interface, all features of the OCD system available through the 10-pin JTAG interface can also be accessed using aWire.

4.2.3. PDI

The Program and Debug Interface (PDI) is an Atmel proprietary interface for external programming and on-chip debugging of a device. PDI Physical is a 2-pin interface providing a bi-directional half-duplex synchronous communication with the target device.

When designing an application PCB, which includes an Atmel AVR with the PDI interface, the pinout shown in the figure below should be used. One of the 6-pin adapters provided with the Atmel JTAGICE3 kit can then be used to connect the JTAGICE3 probe to the application PCB.

Figure 4-5. PDI Header Pinout

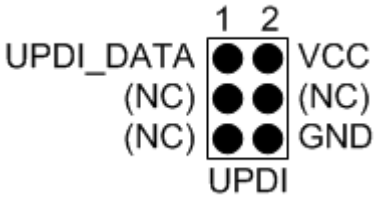


4.2.4. UPDI Physical Interface

The Unified Program and Debug Interface (UPDI) is an Atmel proprietary interface for external programming and on-chip debugging of a device. It is a successor to the PDI 2-wire physical interface, which is found on all AVR XMEGA devices. UPDI is a single-wire interface providing a bi-directional half-duplex asynchronous communication with the target device for purposes of programming and debugging.

When designing an application PCB, which includes an Atmel AVR with the UPDI interface, the pinout shown below should be used. One of the 6-pin adapters provided with the JTAGICE3 kit can then be used to connect the JTAGICE3 probe to the application PCB.

Figure 4-6. UPDI Header Pinout



4.2.4.1. UPDI using JTAGICE3

The JTAGICE3 supports programming and debugging using the UPDI interface. However, the JTAGICE3 has hardware which is not able to drive the 1-wire UPDI interface on pin 3 (AVR header) without driving pin 6 (AVR header) at the same time. For this reason it is recommended to use the 10-pin mini-squid cable for UPDI.

Note: Atmel-ICE and newer tools are capable of direct connection to the UPDI header.

4.2.4.2. UPDI and /RESET

The UPDI one-wire interface can be a dedicated pin or a shared pin, depending on the target AVR device. Consult the device datasheet for further information.

When the UPDI interface is on a shared pin, the pin can be configured to be either UPDI, /RESET, or GPIO by setting the RSTPINCFG[1:0] fuses.

The RSTPINCFG[1:0] fuses have the following configurations, as described in the datasheet. The practical implications of each choice are given here.

Table 4-2. RSTPINCFG[1:0] Fuse Configuration

RSTPINCFG[1:0]	Configuration	Usage
00	GPIO	General purpose I/O pin. In order to access UPDI, a 12V pulse must be applied to this pin. No external reset source is available.
01	UPDI	Dedicated programming and debugging pin. No external reset source is available.
10	Reset	Reset signal input. In order to access UPDI, a 12V pulse must be applied to this pin.
11	Reserved	NA

Note: Older AVR devices have a programming interface, known as "High-Voltage Programming" (both serial and parallel variants exist.) In general this interface requires 12V to be applied to the /RESET pin for the duration of the programming session. The UPDI interface is an entirely different interface. The UPDI pin is primarily a programming and debugging pin, which can be fused to have an alternative function (/RESET or GPIO). If the alternative function is selected then a 12V pulse is required on that pin in order to re-activate the UPDI functionality.

Note: If a design requires the sharing of the UPDI signal due to pin constraints, steps must be taken in order to ensure that the device can be programmed. To ensure that the UPDI signal can function correctly, as well as to avoid damage to external components from the 12V pulse, it is recommended to disconnect any components on this pin when attempting to debug or program the device. This can be done using a 0Ω resistor, which is mounted by default and removed or replaced by a pin header while debugging. This configuration effectively means that programming should be done before mounting the device.

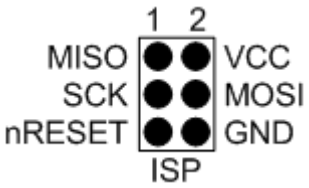
Important: The JTAGICE3 does not support 12V on the UPDI line. In other words, if the UPDI pin has been configured as GPIO or RESET the JTAGICE3 will not be able to enable the UPDI interface.

4.2.5. debugWIRE

The debugWIRE interface was developed by Atmel for use on low pin-count devices. Unlike the JTAG interface, which uses four pins, debugWIRE makes use of just a single pin (RESET) for bi-directional half-duplex asynchronous communication with the debugger tool.

When designing an application PCB, which includes an Atmel AVR with the debugWIRE interface, the pinout shown in the figure below should be used.

Figure 4-7. debugWIRE (SPI) Header Pinout



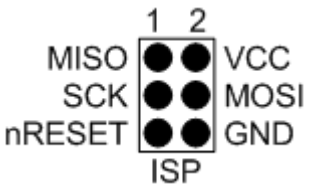
Note: The debugWIRE interface can not be used as a programming interface. This means that the SPI interface must also be available (as shown in Figure 4-8) in order to program the target.

When the debugWIRE enable (DWEN) fuse is programmed and lockbits are un-programmed, the debugWIRE system within the target device is activated. The RESET pin is configured as a wire-AND (open-drain) bi-directional I/O pin with pull-up enabled and becomes the communication gateway between the target and the debugger.

4.2.6. SPI

In-System Programming uses the target Atmel AVR's internal SPI (Serial Peripheral Interface) to download code into the flash and EEPROM memories. It is not a debugging interface. When designing an application PCB, which includes an AVR with the SPI interface, the pinout shown in the figure below should be used.

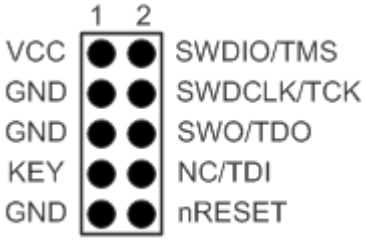
Figure 4-8. SPI Header Pinout



4.2.7. SWD

The ARM SWD interface is a subset of the JTAG interface, making use of TCK and TMS pins. The ARM JTAG and AVR JTAG connectors are however, not pin-compatible, so when designing an application PCB, which uses a SAM device with SWD or JTAG interface, it is recommended to use the ARM pinout shown in the figure below, and use the squid cable or an adapter to map to the AVR pinout used by the JTAGICE3. (There are no adapters available from Atmel to make this mapping.)

Figure 4-9. Recommended ARM SWD/JTAG Header Pinout



The JTAGICE3 is capable of streaming UART-format ITM trace to the host computer. Trace is captured on the TRACE/SWO pin of the 10-pin header (JTAG TDO pin). Data is buffered internally on the JTAGICE3 and is sent over the HID interface to the host computer. The maximum reliable data rate is approx. 1MB/s.

4.3. Atmel OCD Implementations

4.3.1. Atmel AVR UC3 OCD (JTAG and aWire)

The Atmel AVR UC3 OCD system is designed in accordance with the Nexus 2.0 standard (IEEE-ISTO 5001™-2003), which is a highly flexible and powerful open on-chip debug standard for 32-bit microcontrollers. It supports the following features:

- Nexus compliant debug solution
- OCD supports any CPU speed
- Six program counter hardware breakpoints
- Two data breakpoints
- Breakpoints can be configured as watchpoints
- Hardware breakpoints can be combined to give break on ranges
- Real-time program counter branch tracing, data trace, process trace (not supported by Atmel JTAGICE3)

For more information regarding the UC3 OCD system, consult the AVR32UC Technical Reference Manuals, located on www.atmel.com/uc3.

4.3.2. Atmel AVR XMEGA OCD (JTAG and PDI physical)

The Atmel AVR XMEGA OCD is otherwise known as PDI (Program and Debug Interface). Two physical interfaces (JTAG and PDI physical) provide access to the same OCD implementation within the device. It supports the following features:

- Complete program flow control
- One dedicated program address comparator or symbolic breakpoint (reserved)
- Four hardware comparators
- Unlimited number of user program breakpoints (using BREAK)
- No limitation on system clock frequency

4.3.3. Atmel megaAVR OCD (JTAG)

The Atmel megaAVR OCD is based on the JTAG physical interface. It supports the following features:

- Complete program flow control
- Four program memory (hardware) breakpoints (one is reserved)
- Hardware breakpoints can be combined to form data breakpoints

- Unlimited number of program breakpoints (using BREAK) (except ATmega128[A])

4.3.4. Atmel megaAVR/tinyAVR OCD (debugWIRE)

The debugWIRE OCD is a specialized OCD module with a limited feature set specially designed for Atmel AVR devices with low pin-count. It supports the following features:

- Complete program flow control
- Unlimited Number of User Program Breakpoints (using BREAK)
- Automatic baud configuration based on target clock

4.3.5. Atmel tinyX-OCD (UPDI)

The Atmel tinyX-OCD is based on the UPDI (1-wire PDI) physical interface. It supports the following features:

- Complete program flow control
- Two program memory (hardware) breakpoints (one is reserved)
- Unlimited number of program breakpoints (using BREAK)

4.3.5.1. UPDI and /RESET

The UPDI one-wire interface can be a dedicated pin or a shared pin, depending on the target AVR device. Consult the device datasheet for further information.

When the UPDI interface is on a shared pin, the pin can be configured to be either UPDI, /RESET, or GPIO by setting the RSTPINCFG[1:0] fuses.

The RSTPINCFG[1:0] fuses have the following configurations, as described in the datasheet. The practical implications of each choice are given here.

Table 4-3. RSTPINCFG[1:0] Fuse Configuration

RSTPINCFG[1:0]	Configuration	Usage
00	GPIO	General purpose I/O pin. In order to access UPDI, a 12V pulse must be applied to this pin. No external reset source is available.
01	UPDI	Dedicated programming and debugging pin. No external reset source is available.
10	Reset	Reset signal input. In order to access UPDI, a 12V pulse must be applied to this pin.
11	Reserved	NA

Note: Older AVR devices have a programming interface, known as "High-Voltage Programming" (both serial and parallel variants exist.) In general this interface requires 12V to be applied to the /RESET pin for the duration of the programming session. The UPDI interface is an entirely different interface. The UPDI pin is primarily a programming and debugging pin, which can be fused to have an alternative function (/RESET or GPIO). If the alternative function is selected then a 12V pulse is required on that pin in order to re-activate the UPDI functionality.

Note: If a design requires the sharing of the UPDI signal due to pin constraints, steps must be taken in order to ensure that the device can be programmed. To ensure that the UPDI signal can function correctly, as well as to avoid damage to external components from the 12V pulse, it is recommended to disconnect any components on this pin when attempting to debug or program the device. This can be done using a 0Ω resistor, which is mounted by default and removed or replaced by a pin header while

debugging. This configuration effectively means that programming should be done before mounting the device.

Important: The JTAGICE3 does not support 12V on the UPDI line. In other words, if the UPDI pin has been configured as GPIO or RESET the JTAGICE3 will not be able to enable the UPDI interface.

4.3.5.2. TinyX-OCD (UPDI) Special Considerations

The UPDI data pin (UPDI_DATA) can be a dedicated pin or a shared pin, depending on the target AVR device. A shared UPDI pin is 12V tolerant, and can be configured to be used as /RESET or GPIO. For further details on how to use the pin in these configurations, see UPDI Physical Interface.

On devices which include the CRCSCAN module (Cyclic Redundancy Check Memory Scan) this module should not be used in continuous background mode while debugging. The OCD module has limited hardware breakpoint comparator resources, so BREAK instructions may be inserted into flash (software breakpoints) when more breakpoints are required, or even during source-level code stepping. The CRC module could incorrectly detect this breakpoint as a corruption of flash memory contents.

The CRCSCAN module can also be configured to perform a CRC scan before boot. In the case of a CRC mismatch, the device will not boot, and appear to be in a locked state. The only way to recover the device from this state is to perform a full chip erase and either program a valid flash image or disable the pre-boot CRCSCAN. (A simple chip erase will result in a blank flash with invalid CRC, and the part will thus still not boot.) Atmel Studio will automatically disable the CRCSCAN fuses when chip erasing a device in this state.

When designing a target application PCB where UPDI interface will be used, the following considerations must be made for correct operation:

- Pull-up resistors on the UPDI line must not be smaller (stronger) than 10kΩ. A pull-down resistor should not be used, or it should be removed when using UPDI. The UPDI physical is push-pull capable, so only a weak pull-up resistor is required to prevent false start bit triggering when the line is idle.
- If the UPDI pin is to be used as a RESET pin, any stabilizing capacitor must be disconnected when using UPDI, since it will interfere with correct operation of the interface
- If the UPDI pin is used as RESET or GPIO pin, all external drivers on the line must be disconnected during programming or debugging since they may interfere with the correct operation of the interface

4.3.6. ARM Coresight Components

Atmel ARM Cortex-M based microcontrollers implement Coresight™ compliant OCD components. The features of these components can vary from device to device. For further information consult the device's datasheet.

5. Hardware Description

5.1. LEDs

The Atmel JTAGICE3 top panel has three LEDs which indicate the status of current debug or programming sessions.

Table 5-1. LEDs

LED	Function	Description
Left	Target power	GREEN when target power is OK. Flashing indicates a target power error. Does not light up until a programming/debugging session connection is started.
Middle	Main power	RED when main-board power is OK
Right	Status	Flashing GREEN when the target is running/stepping. OFF when target is stopped

5.2. Rear Panel

The rear panel of the Atmel JTAGICE3 houses the Mini-B USB connector.

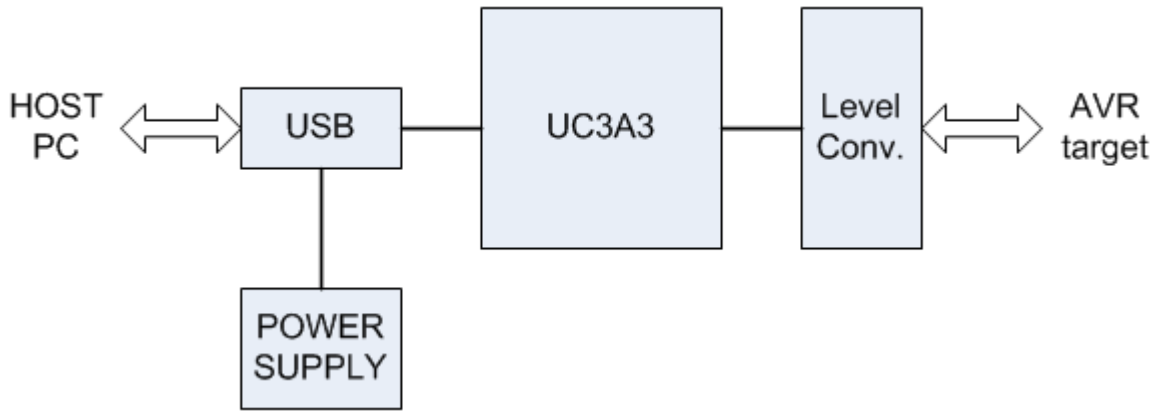
5.3. Bottom Panel

The bottom panel of the Atmel JTAGICE3 has a sticker which shows the serial number and date of manufacture. When seeking technical support, include these details.

5.4. Architecture Description

The Atmel JTAGICE3 architecture is shown in the block diagram in the figure below.

Figure 5-1. JTAGICE3 Block Diagram



5.4.1. Atmel JTAGICE3 Main Board

Power is supplied to the Atmel JTAGICE3 from the USB bus, regulated to 3.3V by a step-down switch-mode regulator. A small amount of power (less than 1mA) is also drawn from the target application by the VTG pin, and is used to power the variable-voltage side of the on-board level converters. At the heart of the JTAGICE3 main board is the Atmel AVR UC3 microcontroller AT32UC3A3256, which runs at between 1MHz and 60MHz depending on the tasks being processed. The microcontroller includes an on-chip USB 2.0 high-speed module, allowing high data throughput to and from the debugger.

Communication between the JTAGICE3 and the target device is done through a bank of level converters that shift signals between the target's operating voltage and the internal voltage level on the JTAGICE3. Also in the signal path are series termination resistors and ESD protection diodes. All signal channels can be operated in the range 1.65V to 5.5V. Maximum operating frequency varies according to the target interface in use.

5.4.2. Atmel JTAGICE3 Target Connectors

The Atmel JTAGICE3 does not have an active probe. A 50-mil IDC cable is used to connect to the target application either directly, or through one of the adapters provided. For more information on the cabling and adapters, see section Assembling the JTAGICE3.

5.4.3. Atmel JTAGICE3 Target Connectors Part Numbers

In order to connect the JTAGICE3 50-mil IDC cable directly to a target board, any standard 50-mil 10-pin header should suffice. It is advised to use keyed headers to ensure correct orientation when connecting to the target, such as those used on the adapter board included with the kit.

The part number for this header is: FTSH-105-01-L-DV-K-A-P from SAMTEC.

6. Software Integration

6.1. Atmel Studio

6.1.1. Software Integration in Atmel Studio

Atmel Studio is an Integrated Development Environment (IDE) for writing and debugging Atmel AVR applications in Windows environments. Atmel Studio provides a project management tool, source file editor, simulator, assembler and front-end for C/C++, programming, emulation, and on-chip debugging.

Atmel Studio or AVR Studio 5.0 or later must be used in conjunction with the Atmel JTAGICE3.

6.1.2. Programming Options

Atmel Studio supports programming of Atmel AVR and Atmel SAM D20 ARM devices using the Atmel JTAGICE3. The programming dialog can be configured to use JTAG, aWire, SPI, PDI, or SWD modes, according to the target device selected.

When configuring the clock frequency, different rules apply for different interfaces and target families:

- SPI programming makes use of the target clock. Configure the clock frequency to be lower than one fourth of the frequency at which the target device is currently running.
- JTAG programming on Atmel megaAVR devices is clocked by the programmer. This means that the programming clock frequency is limited to the maximum operating frequency of the device itself. (Usually 16MHz.)
- AVR XMEGA programming on both JTAG and PDI interfaces is clocked by the programmer. This means that the programming clock frequency is limited to the maximum operating frequency of the device itself. (Usually 32MHz.)
- AVR UC3 programming on JTAG interface is clocked by the programmer. This means that the programming clock frequency is limited to the maximum operating frequency of the device itself. (Limited to 33MHz.)
- AVR UC3 programming on aWire interface is clocked by the programmer. The optimal frequency is given by the SAB bus speed in the target device. The JTAGICE3 debugger will automatically tune the aWire baud rate to meet this criteria. Although it is usually not necessary, the user can limit the maximum baud rate if needed (e.g. in noisy environments).
- SAM D20 ARM programming on SWD interface is clocked by the programmer. The maximum frequency supported by JTAGICE3 is 2MHz. The frequency should not exceed the target CPU frequency *10.

6.1.3. Debug Options

When debugging an Atmel AVR device using Atmel Studio, the 'Tool' tab in the project properties view contains some important configuration options. The options, which need further explanation are:

- **Target Clock Frequency**
  Target clock frequency: Accurately setting the target clock frequency is vital to achieve reliable debugging of an Atmel megaAVR device over the JTAG interface. This setting should be less than one fourth of the lowest operating frequency of your AVR target device in the application being debugged.

  Debug sessions on debugWIRE target devices are clocked by the target device itself, and thus no frequency setting is required. The Atmel JTAGICE3 will automatically select the correct baud rate for communicating at the start of a debug session. However, if you are experiencing reliability

problems related to a noisy debug environment, it is possible to force the debugWIRE speed to a fraction of its "recommended" setting.

Debug sessions on AVR XMEGA target devices can be clocked at up to the maximum speed of the device itself (usually 32MHz).

Debug sessions on AVR UC3 target devices over the JTAG interface can be clocked at up to the maximum speed of the device itself (limited to 33MHz). However, the optimal frequency will be slightly below the current SAB clock on the target device.

Debug sessions on UC3 target devices over the aWire interface will be automatically tuned to the optimal baud rate by the JTAGICE3 itself. However, if you are experiencing reliability problems related to a noisy debug environment, it is possible to force the aWire speed below a configurable limit.

Debug sessions on SAM D20 ARM target devices over the SWD interface can be clocked at up to the target CPU clock *10 (limited to 2MHz maximum).

- **Preserve EEPROM**
  Select this option to avoid erasing the EEPROM during reprogramming of the target before a debug session.
- **Use External Reset**
  If your target application disables the JTAG interface, the external reset must be pulled low during programming. Selecting this option avoids repeatedly being asked whether to use the external reset.

7. Command Line Utility

Atmel Studio comes with a command line utility called that can be used to program targets using the Atmel SAM4S Xplained Pro. During the Atmel Studio installation a shortcut called Atmel Studio 6.2 Command Prompt were created in the Atmel folder on the Start menu. By double clicking this shortcut a command prompt will be opened and programming commands can be entered. The command line utility is installed in the Atmel Studio installation path in the folder Atmel/Atmel Studio 6.2/atbackend/.

To get more help on the command line utility type the command: `atprogram --help`.

8. Advanced Debugging Techniques

8.1. Atmel AVR UC3 Targets

8.1.1. EVTI/EVTO Usage

The EVTI and EVTO pins are not accessible on the Atmel JTAGICE3. However, they can still be used in conjunction with other external equipment.

EVTI can be used for the following purposes:

- The target can be forced to stop execution in response to an external event. If the Event In Control (EIC) bits in the DC register are written to 0b01, high-to-low transition on the EVTI pin will generate a breakpoint condition. EVTI must remain low for one CPU clock cycle to guarantee that a breakpoint is triggered. The External Breakpoint bit (EXB) in DS is set when this occurs.
- Generating trace synchronization messages. Not used by the JTAGICE3.

EVTO can be used for the following purposes:

- Indicating that the CPU has entered debug mode. Setting the EOS bits in DC to 0b01 causes the EVTO pin to be pulled low for one CPU clock cycle when the target device enters debug mode. This signal can be used as a trigger source for an external oscilloscope.
- Indicating that the CPU has reached a breakpoint or watchpoint. By setting the EOC bit in a corresponding Breakpoint/Watchpoint Control Register, breakpoint or watchpoint status is indicated on the EVTO pin. The EOS bits in DC must be set to 0xb10 to enable this feature. The EVTO pin can then be connected to an external oscilloscope in order to examine watchpoint timing.
- Generating trace timing signals. Not used by the JTAGICE3.

8.2. debugWIRE Targets

8.2.1. Software Breakpoints

The debugWIRE OCD is drastically scaled down when compared to the Atmel megaAVR (JTAG) OCD. This means that it does not have any program counter breakpoint comparators available to the user for debugging purposes. One such comparator does exist for purposes of run-to-cursor and single-step operations, but user breakpoints are not supported in the hardware.

Instead, the debugger must make use of the Atmel AVR BREAK instruction. This instruction can be placed in FLASH, and when loaded for execution it will cause the AVR CPU to enter stopped mode. To support breakpoints during debugging, the debugger must insert a BREAK instruction into FLASH at the point at which the users requests a breakpoint. The original instruction must be cached for later replacement. When single stepping over a BREAK instruction, the debugger has to execute the original cached instruction in order to preserve program behavior. In extreme cases, the BREAK has to be removed from FLASH and replaced later. All these scenarios can cause apparent delays when single stepping from breakpoints, which will be exacerbated when the target clock frequency is very low.

It is thus recommended to observe the following guidelines, where possible:

- Always run the target at as high frequency as possible during debugging. The debugWIRE physical interface is clocked from the target clock.
- Try to minimize on the number of breakpoint additions and removals, as each one require a FLASH page to be replaced on the target

- Try to add or remove a small number of breakpoints at a time, to minimize the number of FLASH page write operations
- If possible, avoid placing breakpoints on double-word instructions

9. Firmware Upgrade

For information on how to upgrade the firmware, see the Atmel Studio user guide.

10. Release History and Known Issues

10.1. What's New in Major Version 3

Firmware version 3 adds SAM support

When installing Atmel Studio 6.1 SP2, the JTAGICE3 will receive a major firmware increment from version 2 to version 3. While version 2 firmware enumerates as a Jungo USB tool in the device manager, version 3 appears as a HID device (with a new USB PID), much like the EDBG included in Xplained Pro series of evaluation kits.

The new JTAGICE3 firmware brings support for all Atmel SAM ARM Cortex-M based microcontrollers to the JTAGICE3, while preserving all of the AVR functionality.

As of version 3.11, the JTAGICE3 also supports serial ITM trace capture.

Note:
After upgrading to firmware version 3, the JTAGICE3 will NOT be recognized by previous versions of Atmel Studio. In order to use the JTAGICE3 with an older version of Atmel Studio, it must first be downgraded back to version 2 firmware.

The `atfw.exe` command-line firmware upgrade utility must be used with the `-a archive-name` switch to give the utility the path to the version 2 firmware zip file. After the downgrade, older versions of Atmel Studio will work as expected.

The `atfw.exe` command-line upgrade utility can be downloaded from the Atmel website or from the Atmel Gallery. The Atmel Gallery download contains `atfw.exe` and the needed firmware version.

For information on connecting the JTAGICE3 to SAM devices using SWD, see Connecting to an SWD Target.

10.2. Firmware Release History

Table 10-1. Public Firmware Revisions

Firmware version (decimal)	Date	Relevant changes
3.34	29.09.2016	Added support of UPDI interface (tinyX devices). Made USB endpoint size configurable.
3.30		Fixed target power led issue. Fixed issue related to JTAG daisy chains with more than 64 instruction bits.
3.23		Fixed oscillator calibration command. Improved debugWIRE reliability.
3.18		Support for all SAM ARM Cortex-M based microcontrollers. ITM trace capture support added.
3.7		Support for SAM D20 ARM Cortex-M0+ based microcontrollers

Firmware version (decimal)	Date	Relevant changes
2.15		Minor bug fixes
2.10		Minor bug fixes
1.34		JTAG daisy-chain support. Improved external- and software-reset handing. Support for high SUT values on Atmel AVR XMEGA devices. aWire auto-baud calculation improvements. Chip erase timeout corrected for newer Atmel AVR XMEGA devices. Fixed XMEGA flash page programming error (seen at low voltages). Improved debugWIRE single-stepping performance.
1.22		Improved status LEDs
1.17		Improved LED behavior. Improved debugWIRE reliability. Improved aWire speed.
1.08		First release of JTAGICE3

10.3. Known Issues Concerning the Atmel JTAGICE3

10.3.1. Atmel AVR XMEGA OCD Specific Issues

- For the ATxmegaA1 family, only revision G or later is supported
- PDI might not work at certain frequencies if there is much undershoot on the PDI clock signal. On some target board configurations the PDI clock has shown a fair amount of undershoot. This undershoot might result in the Atmel JTAGICE3 not being able to communicate with the target. To get around this problem the PDI clock frequency should be lowered. This can be done on the Interface settings tab in the programming dialog or on the Tool tab in the project Properties of a debug project. If a higher PDI clock frequency is required some kind of termination must be added to the PDI clock line. For example adding a small capacitor (in the order of 100pF) on the target board between the PDI clock and V_{CC} has proven to be effective. Adding a series resistor is not effective unless it is added to the JTAGICE3 main board.

10.3.2. Atmel megaAVR OCD and Atmel tinyAVR OCD Specific Issues

- Cycling power on ATmega32U6 during a debug session may cause a loss of contact with the device

11. Product Compliance

11.1. RoHS and WEEE

The Atmel JTAGICE3 (kit ATJTAGICE3) and its accessories are manufactured in accordance to both the RoHS Directive (2002/95/EC) and the WEEE Directive (2002/96/EC).

11.2. CE and FCC

The Atmel JTAGICE3 unit has been tested in accordance to the essential requirements and other relevant provisions of Directives:

- Directive 2004/108/EC (class B)
- FCC part 15 subpart B
- 2002/95/EC (RoHS, WEEE)

The following standards are used for evaluation:

- EN 61000-1 (2007)
- EN 61000-3 (2007)
- FCC CRF 47 Part 15 (2008)

The Technical Construction File is located at:

Atmel Norway

Vestre Rosten 79

7075 Tiller

Norway

Every effort has been made to minimize electromagnetic emissions from this product. However, under certain conditions the system (this product connected to a target application circuit) may emit individual electromagnetic component frequencies, which exceed the maximum values allowed by the above mentioned standards. The frequency and magnitude of the emissions will be determined by several factors, including layout and routing of the target application with which the product is used.

12. Revision History

Doc. Rev.	Date	Comments
42634B	10/2016	Added UPDI interface and updated Firmware Release History
42634A	11/2015	Initial document release

Atmel Corporation 1600 Technology Drive, San Jose, CA 95110 USA **T:** (+1)(408) 441.0311 **F:** (+1)(408) 436.4200 | **www.atmel.com**