Diagnostics

series

SIGNUM SYSTEMS CORPORATION

Emulator Diagnostic Utility

User Manual

SIGNUM SYSTEMS

COPYRIGHT NOTICE

DISCLAIMER

Signum Systems makes no representations or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Also, Signum Systems reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation of Signum Systems to notify any person or organization of such revision or changes.

WARRANTY

Signum Systems warrants to the original purchaser that this product is free of defects in material and workmanship and performs to applicable published Signum Systems specifications for a period of SIX MONTHS from the date of shipment. If defective, the product must be returned to Signum Systems, prepaid, within the warranty period, and it will be repaired or replaced (at our option) at no charge. Equipment or parts which have been subject to misuse, abuse, alteration, neglect, accident, unauthorized installation or repair are not covered by warranty. This warranty is in lieu of any other warranty expressed or implied. IN NO EVENT SHALL SIGNUM SYSTEMS BE LIABLE FOR CONSEQUENTIAL DAMAGES OF ANY KIND. It is up to the purchaser to determine the reliability and suitability of this product for his particular application.

SIGNUM
SYSTEMS
1211 FLYNN RD., UNIT #104
CAMARILLO, CA 93012, U.S.A
PHONE 805 • 383 • 3682
WWW.SIGNUM.COM

Diagnostics

series

Introduction

EmuDiag allows you to test the connection between your computer and the emulator as well as update the emulator firmware. Additionally, the program can diagnose and test the JTAG connection to your target board. Even though this document refers mainly to Signum JTAGjet emulators, it is also applicable to the ADM-51 emulator. A rudimentary JTAG design guidelines and troubleshooting section is included to help you solve problems when connecting to the CPU and determining a malfunction of the emulator.

Installation

Almost all Signum software products — debugger drivers, flashers etc. — include EmuDiag.exe. So does Chameleon Debugger. A shortcut to EmuDiag is installed in the Windows Start menu for a particular program or driver. For example, EmuDiag that comes with the Code Composer Studio drivers for JTAGjet can be started from the Start menu entry as shown in Figure 1

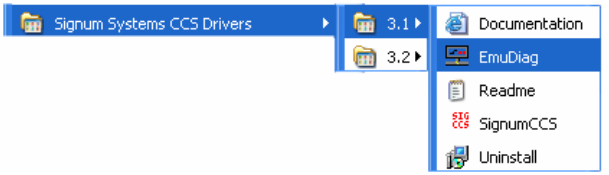


FIGURE 1 Starting EmuDiag.

If EmuDig is not automatically installed, the user can install the utility from the *Signum Systems Development Tools for MS Windows* CD. Insert the disk into your CD-ROM drive, and from the Master Setup screen, select JTAGjet Utilities. Double-click EmuDiag (FIGURE 2FIGURE 2FIGURE 2).

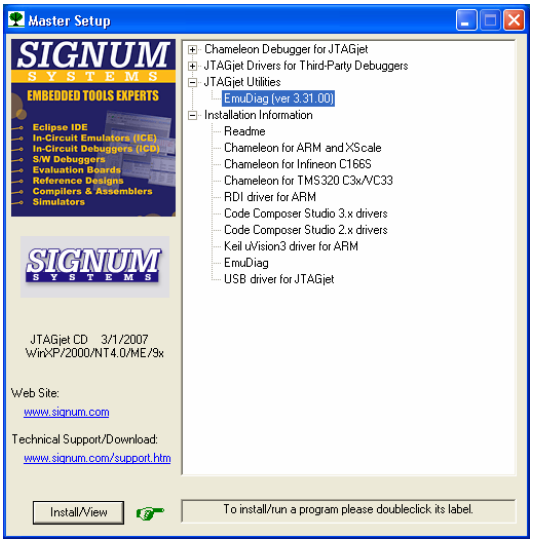


FIGURE 2 Master Setup dialog box.

To complete the installation process, follow the on-screen instructions. The Installation Information section of the dialog box (Figure 2) provides access to pertinent documentation.

The latest version of EmuDiag along with the current documentation may be downloaded free of charge at www.signum.com/support.htm.

The EmuDiag package includes the latest version of system-level USB drivers for the JTAGjet and ADM51 emulators. Drivers older than those in the package are updated during the installation process.

Connecting to the Emulator

When starting, EmuDiag prompts you for the method of connecting to the emulator (Figure 3).

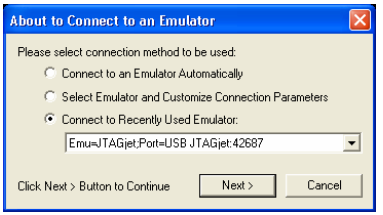


FIGURE 3 Selecting the method of connecting to the emulator.

- The **Connect to Emulator Automatically** method searches for emulators on USB ports. Click the Next button to connect automatically. If there is more than one emulator physically connected, you will need to select one.
- The **Select Emulator and Customize Connection Parameters** method allows you to select your emulator and customize the connection parameters. Click Next to display a list of the available emulators (Figure 4).

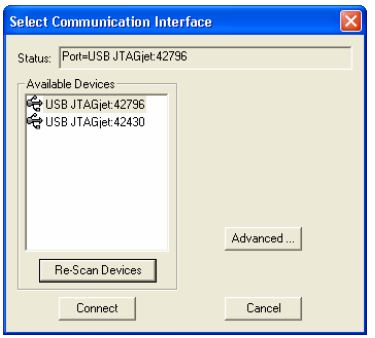


FIGURE 4 Choosing the emulator to connect to.

Select the emulator and click Connect.

The Advanced button allows the user to customize certain connection parameters. It should not be used without guidance from a Signum Technical Support engineer.

- The **Connect to Recently Used Emulator** method is designed to reconnect to the emulator with the settings of an earlier successful connection (Figure 5).

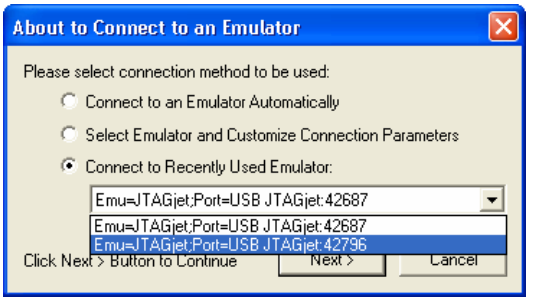


FIGURE 5 Reapplying the parameters of an earlier connection.

Select the emulator and click the Next button to establish the connection. If the emulator with the selected serial number is not connected to the USB port, EmuDiag will attempt to connect to another emulator of the same type. If there are multiple emulators of the same kind to choose from, user intervention becomes necessary.

EmuDiag begins establishing communication with the emulator using the method and parameters you have selected. Typically, the initial EmuDiag screen that appears after connection has been established looks like this (Figure 6).

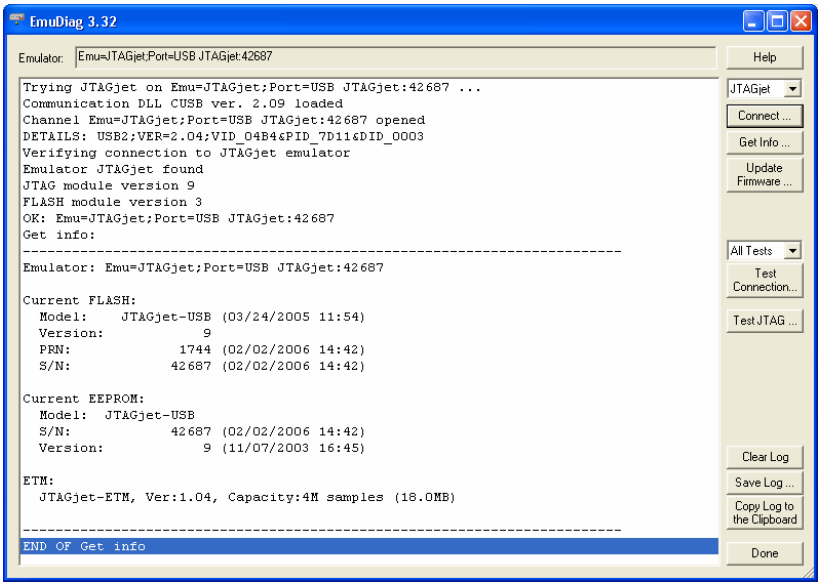


FIGURE 6 EmuDiag has just connected to an emulator.

The key items displayed in the log window are as follows.

Title: **EmuDiag 3.32**

The EmuDiag version is always displayed in the title bar.

Emulator: **Emu=JTAGjet;Port=USB JTAGjet:42687**

The name of the currently connected emulator (JTAGjet) and its serial number (42678) is displayed above the log window.

DETAILS: USB2;VER=2.04;VID_04B4&PID_7D11&DID_0003

USB port connection details:

USB2 - USB 2.0 protocol. You may see USB1.1 here.

VER=2.04 - System level USB driver (SigUSB.sys) version.

JTAGjet is a high-speed USB2 device (480MHz USB port), but it also works with the slower USB1.x ports. The mulation speed for USB2.0 ports is in some cases orders of magnitude faster than for USB1.1 ports. It is strongly recommended then that you invest in an USB2.0 card (PCI or PCMCIA) if only USB 1.1 ports available on your system.

JTAG module version 9

The version of the hardware module:

9 - HW version of the JTAG controller. Newer JTAG module versions may be associated with newer JTAGjet models.

JTAGjet-ETM, Ver:1.04, Capacity: 4M samples (18.0MB)

These details are displayed only for the JTAGjet-ETM model.

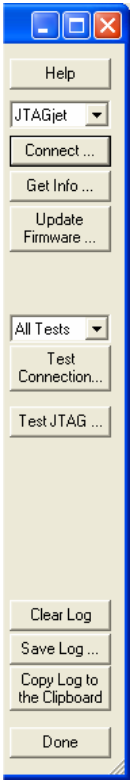
1.04 - HW version of the JTAGjet-ETM module.

Capacity - trace memory capacity (4 millions of samples, 18M bytes of RAM). These 18M needs to be read by debugger. In practice, the JTAGjet-ETM requires USB2.0 port.

In case of any problems or errors, please email the complete EmuDiag log information to Signum Technical Support.

EmuDiag Buttons

After connecting to the emulator, the following fields appear in the right side of the EmuDiag window (Figure 7).



FIELD	DESCRIPTION
Help	Displays the EmuDiag PDF help file.
JTAGjet/ADM51/	Selects the emulator you would like to connect to.
Connect	Displays the connection dialog box and version summary.
Get Info	More detailed description of modules.
Update Firmware	Browse for the UPG file and update the JTAGjet firmware.
All Tests/Selection	Determines which connection tests will be executed.
Test Connection	Executes the connection tests.
Test JTAG	Setup and test the JTAG connection to the target.
Clear Log	Clears the messages in the log window.
Save Log	Saves a log with messages to a text file.
Copy Log to the Clipboard	Copies the current log window to the clipboard.
Done	Terminates EmuDiag.

FIGURE 7 EmuDiag controls

The Test Connection, Test JTAG, and Update Firmware functionality (see the Field column in the table above) is described further in the text.

Testing USB Connection

Click the Test Connection button to execute reliability and performance tests. These tests examine the USB connection between the PC and the JTAGjet emulator. The target does not have to be connected to the JTAG port for these

tests to run. The reliability test executes a series of patterns on the USB channel and checks the responses from the JTAGjet. The performance test measures the packet transfer speed to and from the emulator.

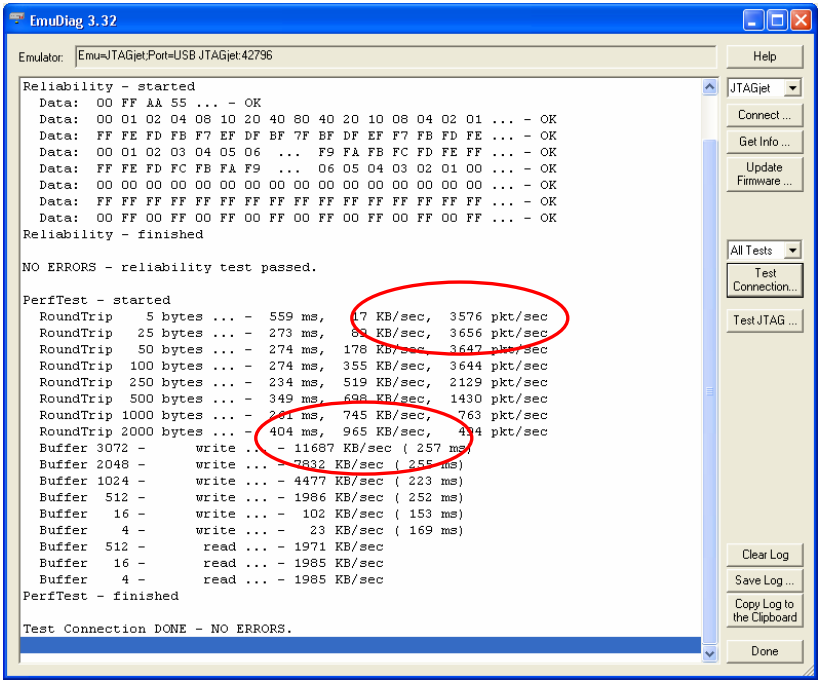


FIGURE 8 Measuring packet transfer spo

Provided that the emulator hardware functions normally, reliability tests on the USB port will never show errors. This is because the USB protocol includes check-sum and will re-transmit corrupted packets and silently recover from any hardware related data errors.

The performance test evaluates most of the settings affecting the performance of the debug process and and overall experience. These settings, displayed in the EmuDiag window, are explained below. Note that higher values correspond to better performance.

RoundTrip 5 bytes	**3576 pkt/sec**	more than 3500 short packets can be exchanged with the JTAGjet.
Buffer 3072 – write	**11687 KB/sec**	one-way write speed for larger 3K packets.

When using USB 2.0 ports, you should see similar values on your system. These values usually do not depend on the type or speed of the CPU, but on the type and model of the USB controller in your computer. Some brand-name add-on PCI USB 2.0 cards provide excellent performance.

If Selection is used instead All Tests (above the button), it is possible to select individual tests to be run. To select individual tests, choose Select form the drop-down list located above the Test Connection button.

The USB transfer time histogram test runs for 20 seconds, during which it captures the USB response time and displays a histogram of individual packet transfer time (Figure 9).

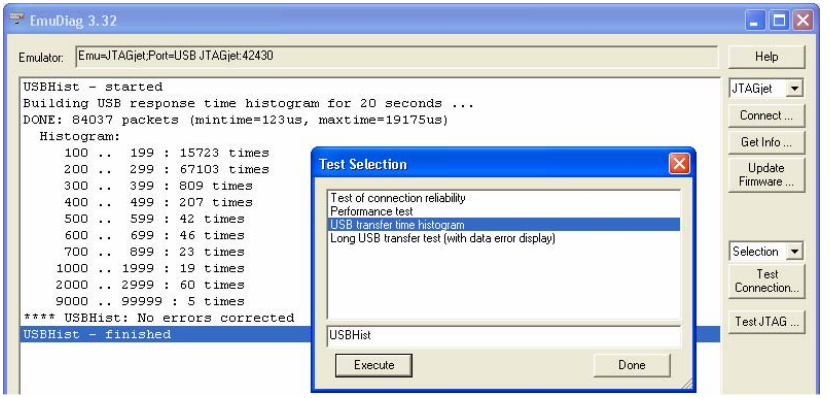


FIGURE 9 Executing the USB transfer histogram test.

As the JTAGjet always responds immediately, all the differences in packet transfer times are caused by Windows system level scheduling. In our example, the majority of packets are transferred in less than 300µs. This corresponds to more than 3300 packets/sec. The longest recorded USB packet transfer time in this example is 19ms, which most probably was caused by Windows turning the

EmuDiag process off. A large number of packets that travel long indicate an overloaded CPU or physical memory running out.

Configuring and Testing JTAG

When the Test JTAG button is pressed, the JTAG Parameters tab in the New Configuration window appears (Figure 10).

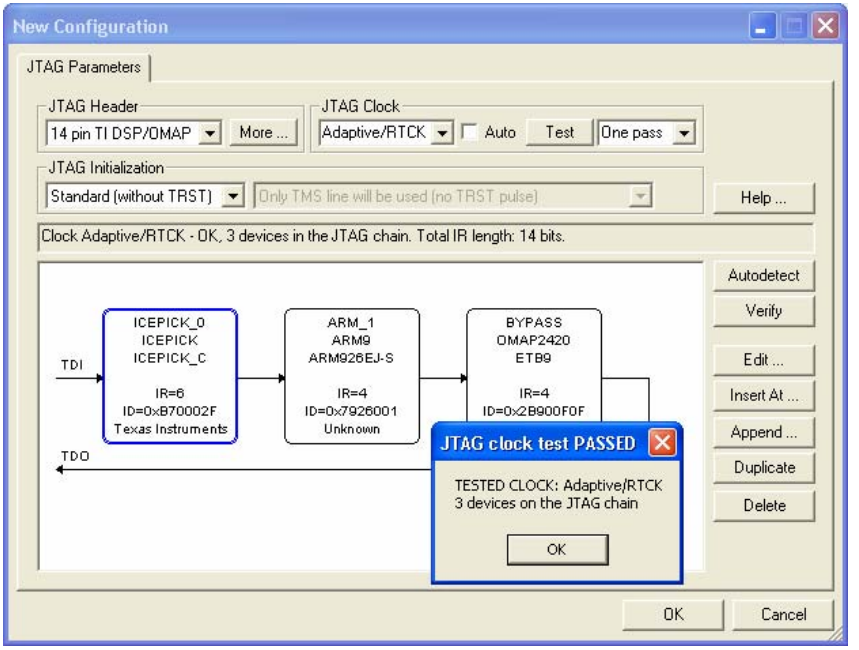


FIGURE 10 JTAG Parameters tab.

The graphics in the tab shows an auto-detected DaVinci processor (DM6446). The "JTAG clock test PASSED message" notifies the user of the results of pressing the Test button. A list of the JTAG devices on the JTAG chain (from TDI to TDO) is displayed as a result of pressing the Autodetect button.

The following sections describe EmuDiag's JTAG-related functionality in detail.

The Graphical JTAG configuration module is available independently of EmuDiag — other software modules, such as Code Composer Studio drivers, also incorporate it. All future JTAGjet drivers as well as Chameleon debugger will be integrated with this module.

Selecting JTAG Header

The selection of the JTAG header (sometimes referred to as header, probe or connector) should be accomplished through the JTAG Header drop-down list (Figure 11).

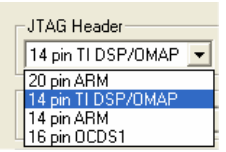


FIGURE 11 selecting the JTAG header

Select the header that matches your adapter. Illustrations and detailed description of all available JTAGjet adapters can be found in the *JTAG Probes for Signum Emulators* document available in PDF format at www.signum.com.

Table 1 matches Signum adapters with the corresponding JTAG header

SIGNUM ADAPTER PART NUMBER	JTAG HEADER SELECTION
ADA-JTAG-ARM20 **ADA-JTAG-ARM20-LV**	20 pin ARM
ADA-JTAG-ARM14	14 pin ARM
ADA-JTAG-TMS **ADA-JTAG-TMS-LV** **ADA-JET-CTI20** **ADA-ISO-TI14**	14 pin TI DSP/OMAP

SIGNUM ADAPTER PART NUMBER	JTAG HEADER SELECTION
ADA-JET-ETM (JTAGjet-ETM splitter)	20 pin ARM or 14 pin TI DSP/OMAP
JTAGjet-ETM (38 pin)	20 pin ARM

TABLE 1 Signum JTAG adapters.

JTAG tests will work even if a wrong header is selected, because the standard JTAG signals (TCK, TMS, TDI, TDO and nTRST) are always available. However, the selected header defines which additional signals can be used, system reset being the most important. If you have selected a wrong type of the header, the debugger may report that CPU is being held in reset, while in reality there is no reset on the header. The software may also try to drive pins that either do not exist or have unexpected assignments.

Selecting and Testing the JTAG Clock Frequency

The desired JTAG clock speed can be selected from a drop-down list (Figure 12).

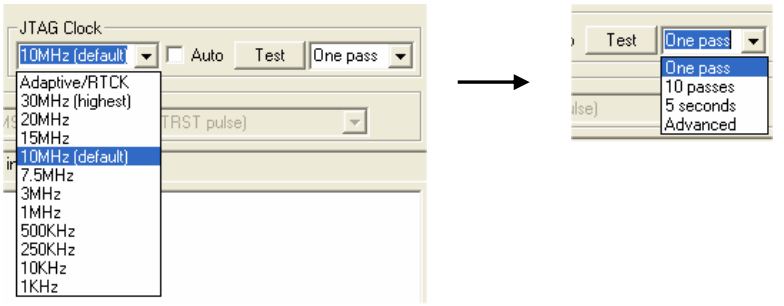


FIGURE 12 Selecting the speed of the JTAG clock.

After selecting the clock frequency, press the Test button, deciding how many passes of the test should be run. Specific bit patterns are scanned on the TDI

pin (to the device). The output on the TDO pin (from the device) are captured and compared with the TDI input. If the two patterns do not match, an error is displayed (Figure 13).

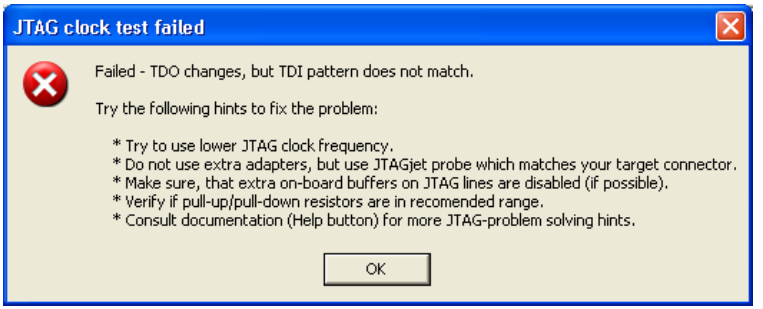


FIGURE 13 Mismatches between TDI and TDO reported during a test.

The error in Figure 13 occurs when the device responds, but the JTAG scanning process is not stable. To fix the problem, select a slower clock, or use the Adaptive/RTCK option, and re-run the test.

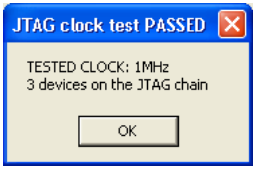


FIGURE 14 Reporting a stable clock.

Reliable performance of the selected clock (1MHz in this example) is reported as shown in Figure 14.

Instead of selecting a preset value for the JTAG frequency, you can enter it from the keyboard as an arbitrary value (e.g., 2.1 MHz). The JTAGjet supports integral frequencies only. Hence, the input value will be rounded down to the nearest available frequency that does not exceed the value you specified. In our example, the resultant frequency will be 2MHz.

Checking the Auto check box makes the software try to auto-detect the JTAG clock frequency by running a series of patterns on the JTAG at different clock speeds (Figure 15).

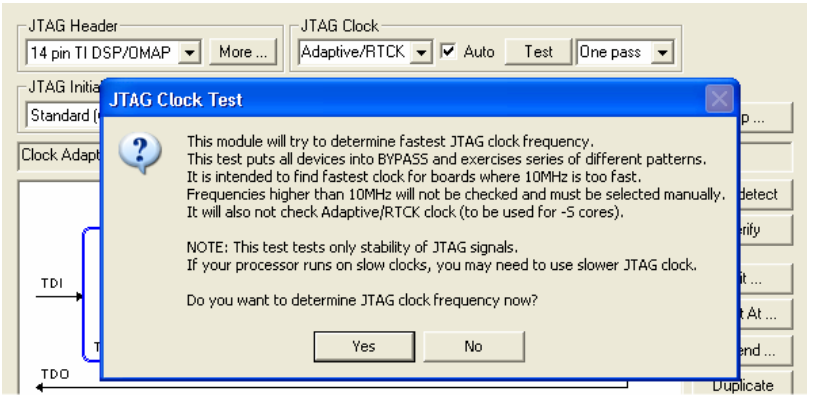


FIGURE 15 Determining the highest working frequency.

The highest JTAG clock frequency allowed is determined and set. In our example, it is 1.875 MHz (Figure 16).

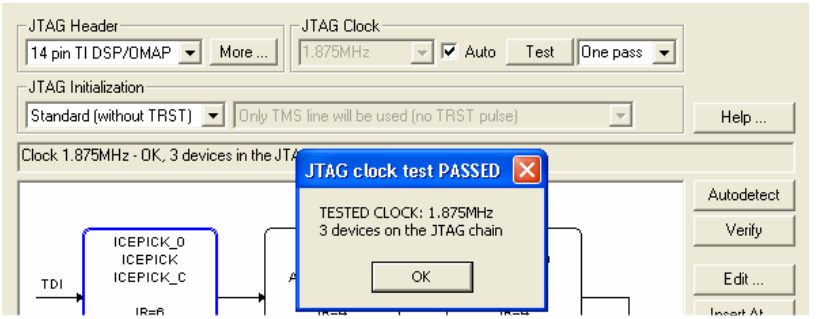


FIGURE 16 The maximum JTAG speed has been detected.

In the target used during our test, the PLL was not initialized. The CPU was running with a slow clock, which was the reason why the faster JTAG clock settings failed.

As the message in Figure 15 indicates, that automatic detection will try neither clock frequencies higher than 10MHz (15, 20 or 30MHz) nor Adaptive/RTCK. You must select these frequencies manually, after which you should test their validity.

ADAPTIVE CLOCK

The Adaptive/RTCK JTAG Clock should be selected for all synthesized cores denoted by the –S suffix after the core's name, such as the ARM7TDMI-S or ARM926EJ-S. This is the most optimal setting for these cores that allows them to lower the CPU speed and conserve battery power without losing the communication link to the emulator. You should make sure that the RTCK signal on the JTAG probe is connected to the device.

For tips covering a number of JTAG-related tips, please refer to the section titled *Basic JTAG Debugging Guidelines* on p. 24.

JTAG Chain Autodetection and Verification

It is a good idea to try to auto-detect the devices attached to the JTAG chain after selecting and testing the JTAG clock. Click the Autodetect button. In most cases, you will see just a single ARM CPU. More complex cases like, for instance, the STR912 processor with its 3 devices on the JTAG chain, can be also auto-detected:

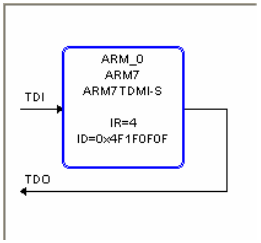


FIGURE 17 ARM7TDMI-S core auto-detected.

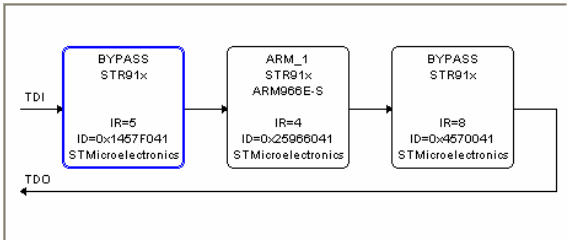


FIGURE 18 STR912 processor auto-detected.

Auto-detection is based on JTAG scans and JEDEC JTAG IDs read from the JTAG. IR values show the length of the JTAG instruction register for that device.

In the first case, the ID is 0x4F1F0F0F, which is ARM-assigned ID for the ARM7TDMI-S core (Figure 17). Note that manufacturer of the chip is unknown. This is expected — the majority of CPU manufacturers do not change the default ARM7 core IDs.

In the second case, all JTAG devices have manufacturer-assigned unique IDs appended to their names. In our example, it is 041, the manufacturer code for ST Microelectronics. EmuDiag has a large built-in database containing devices and their ID numbers. Exact processor model (STR911, STR912) cannot be determined from JTAG IDs and is not shown. Please note that the middle device appears as the ARM966E-S. This is because a part of that device's ID is defined by ARM – only the manufacturer ID was changed by ST.

Auto-detection will always detect the number of the devices on the JTAG chain and the total number of bits. However, if the designer of the CPU did not implement IDs in all the devices available on the JTAG inside of the chip, detecting the device's type and the number of bits in it might be impossible.

Different JTAG devices are oftentimes shared among different CPU models, which may result in auto-detecting the name of the core or the device incorrectly (Figure 19).

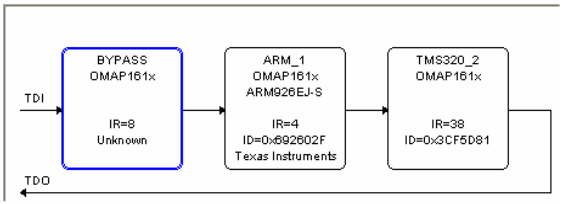


FIGURE 19 Incorrect auto-detection.

Above image shows OMAP161x, but in reality the processor is the OMAP5912. Simply, OMAP161x and OMAP5912 use the same devices with the same JTAG IDs. Please note that the leftmost device (IR=8) does not have an ID. Only the device in the center (the ARM926EJ-S) has one: 02F denoting Texas Instruments as the manufacturer.

In general, most of Texas Instruments DSP processors do not implement JTAG IDs, and very often you may see pictures as follows:

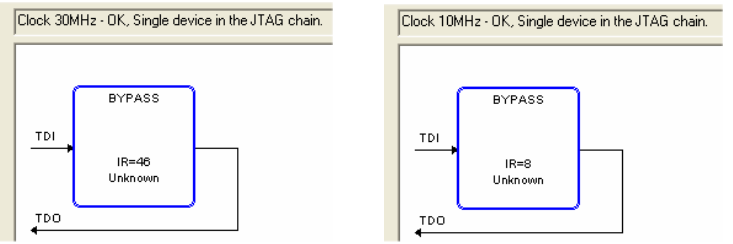


FIGURE 20 Devices without IDs.

The devices in Figure 20 do not have JTAG IDs. Only one of them is detected, thus its IR is known. When there are multiple devices on the JTAG, the following may occur (Figure 21):

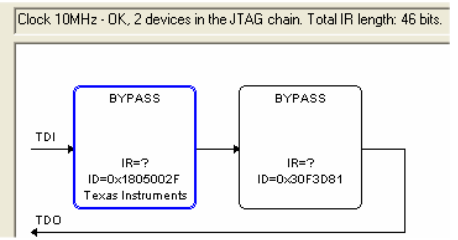


FIGURE 21 Multiple-device JTAG chain with unknown IRs.

EmuDiag is unable to determine the IR for each device. Consult the CPU documentation to determine the order of the devices. Note that EmuDiag categorizes an unknown device as BYPASS.

Table 2 lists the IR length of several common devices and cores.

JTAG DEVICE OR CORE	IR BITS
ARM7, ARM9, OMAP, TMS470	4
ARM11, OMAP2, MPCore	5
Cortex, OMAP3	4
ETB (Embedded Trace Buffer)	4
XScale (Intel)	4 or 5
Older TI's TMS320 DSP: C24x, C54x, C620x, C670x	8

JTAG DEVICE OR CORE	IR BITS
Newer TI's TMS320 DSP: C621x, C671x, C672x	46
Newest TI's TM320 DSP: C28x, C55x, C64x, C64x+	38
ICEPICK (Texas Instruments)	2 or 6

TABLE 2 The IR length of selected processors.

The OMAP and DaVinci processors comprise a few JTAG devices and cores, often in various combinations. Some of these CPUs, such as DaVinci DM6446, have alternative boot modes. The list of the devices visible on the JTAG chain may vary, depending on the mode selected using the board switches.

Manual JTAG Chain Editing

In most cases, EmuDiag will auto-detect all the devices on the JTAG chain correctly, even though the types of some devices may not be available (denoted as BYPASS). Use the Edit button to modify the properties of a BYPASS device to which you want to connect. In addition, you may define the entire JTAG chain manually using Insert At, Append, Duplicate and Delete buttons.

The JTAGjet drivers and Chameleon debugger may not have all information available for all devices. It may be necessary then to re-define the JTAG geometry either in the driver and debugger specific dialog boxes or in the configuration files.

JTAG configuration via EmuDiag should only be used as a diagnostic and verification aid. Future releases of the Signum drivers and Chameleon debugger will use a unified and consistent method of defining JTAG chain geometry and its parameters.

Advanced JTAG Topics

The advanced JTAG-related functionality of EmuDiag is intended to be used mainly for technical support purposes, such as remote connection sessions, in which Signum Systems technical personnel participates. Normally, ordinary tasks diagnostic tasks do not require EmuDiag's advanced features. If they need to be used for a board or CPU, refer to the board documentation for detailed instructions.

Non Standard JTAG Initialization

When a debugger or driver connects to the CPU for the first time (or after power-cycling the target) the CPU's JTAG state machine needs to be reset in one of the two following methods prescribed by the by JTAG standard:

- By pulsing the TCK pin at least 5 times, while holding the TMS pin.
- By using the TRST pin (driving it low and high).

As the TRST line is not always available, the TCK method is set as a default.

I most cases, the standard method is resets the JTAG correctly. Certain new processors, example of which is the DaVinci DM6446, implement a special JTAG controller (ICEPICK) to define the JTAG geometry dynamically. Depending on the state of the EMU0 and EMU1 pins, you may see one or three devices when auto-detecting the DM6446 processor (Figure 22).

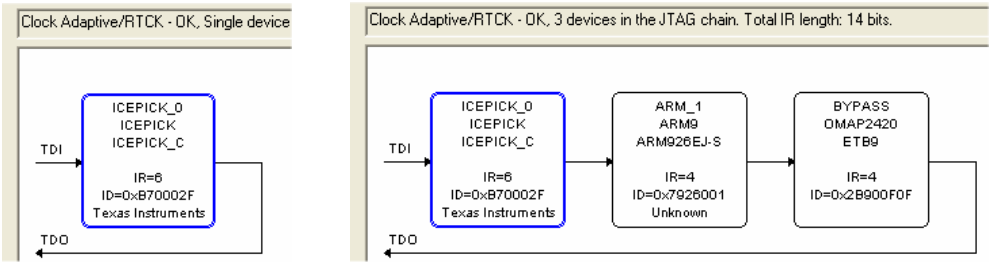


FIGURE 22 Auto-detecting the DaVinci DM6446 processor. Note the use of Adaptive/RTCK clock.

In both cases, the ICEPICK_C device appears as the first one on the JTAG chain. Unfortunately, the ICEPICK reverts the JTAG geometry to the default state after every JTAG reset or power-cycle of the board. (The default state is selectable with the EMU0/EMU1 pins, as described in the processor's documentation.) To ensure that regardless of the initial state of the EMU0/EMU1 pins, the driver or debugger will always "see" the same geometry, select one of the pre-defined JTAG initialization schemes (Figure 23).

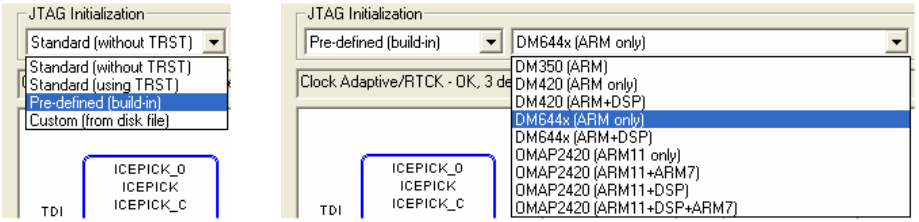


FIGURE 23 Selecting a predefined configuration.

The selection of the "DM644x (ARM only)" configuration results in auto-detecting the JTAG chain shown in Figure 24. There are two devices in it, both shown as BYPASS.

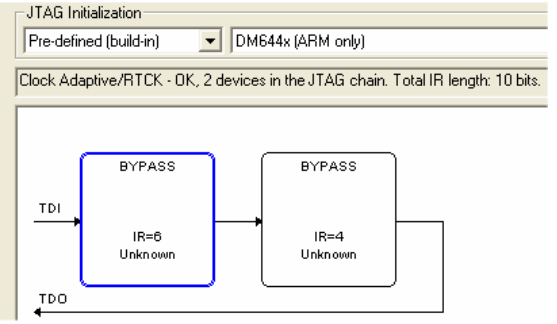


FIGURE 24 JTAG chain auto-detected when a predefined DM64x (ARM only) configuration.

The first one is ICEPICK and the other one is ARM. Their JTAG IDs are not available, because they are read after the standard JTAG reset. Nonstandard standard initialization results in unavailability of the IDs.

The "Custom (from disk file)" option allows you to browse for a file with low-level JTAG initialization commands. Please contact Signum Technical Support if you require special scans of your JTAG chain.

Advanced JTAG Control

Advanced control of the JTAG, including individual bit access and other aspects of the JTAG connection is available through the Advanced JTAG Configuration dialog box. Click the More button located in the "JTAG header" group (Figure 10) to open it, as shown in Figure 25.

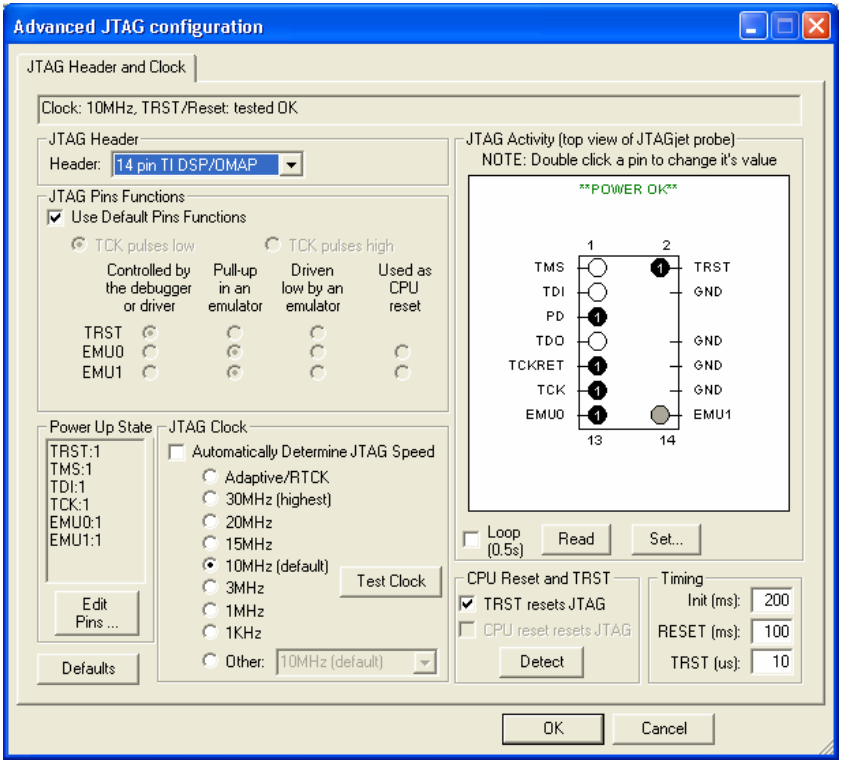


FIGURE 25 The Advanced JTAG Configuration.

The dialog box displays the pin-out of the selected JTAG header. The black circles symbolize pins with logical value 1 on them. White circles symbolize pins with logical value 0. Recently changed pins are denoted with gray circles. You can modify individual pin values by double-clicking the corresponding circles and observing values on your target. The feature is useful for basic validation of the JTAG signals, especially when the JTAG appears not working at all. Keep in mind that on some CPUs, changes on such pins as EMU0 or EMU1, may re-define the CPU boot mode.

The interface also allows you to re-assign the functions of TRST, EMU0, and other JTAG pins. Press the Defaults button to reset all settings to their default values.

Advanced JTAG Clock Tests

To execute JTAG scans of increased complexity, select Advanced form the drop-down list next to the Test button (Figure 12). The feature, which generates a sequence of JTAG actions in a loop, requires expert knowledge of JTAG. It is designed to enable the user to observe the timings and the quality of the resulting JTAG signals with an oscilloscope.

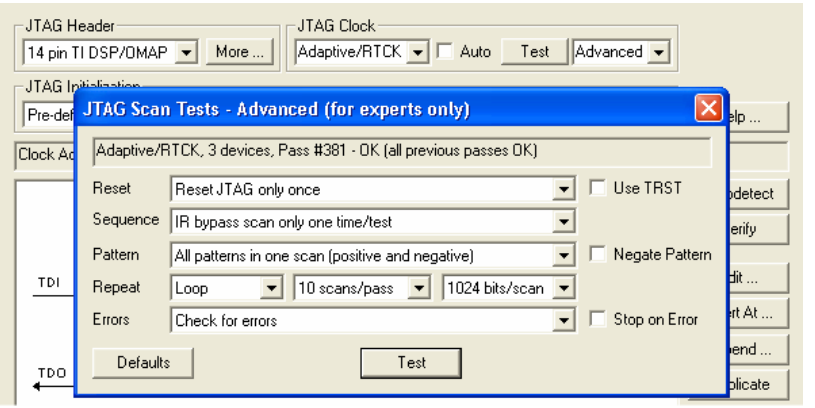


FIGURE 26 Configuring a JTAG scan test.

One "pass" corresponds to a sequence of scans. A scan is a bypass pattern of the specified number of bits. By default, scanning does not stop if errors are detected. However, the output JTAG signals (TCK/TMS/TDI) are generated, so the test can be used to diagnose the problem when some JTAG lines seem to be broken or a JTAG clock does not work on certain boards. The Errors field allows you to specify that the EMU0/DBGRQ pin will be pulsed if an error occurs. This allows the pin to become a trigger for a logic analyzer or scope.

If the fastest possible TDI change on each TCK edge is desired, use the settings shown in Figure 27. The long (100 scans × 4K bits each) passes and display refresh rate of 0.5 second make the TCK ,TDI and TDO pulses on the scope very stable.

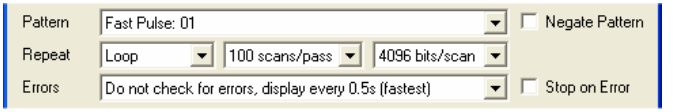


FIGURE 27 Maximizing the speed of the DI change on the TCK edge.

Command-Based JTAG Tests

This feature should be used with the guidance from a Signum engineer. In the main JTAG Parameters dialog box click Cancel. The JTAG Command Entry dialog box opens, enabling you to execute low-level JTAG commands.

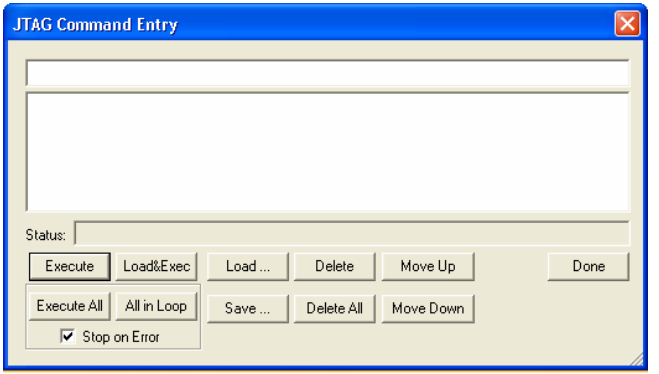


FIGURE 28 JTAG Command Entry dialog box.

Basic JTAG Debugging Guidelines

In general, when building your own board, follow the JTAG design guidelines and observe the, pull-up and pull-down values as defined by the CPU or DSP manufacturer. Searching the Texas Instruments' www.ti.com and ARM's www.arm.com web sites for the keyword "JTAG" will provide you with a wealth of JTAG-related PDF files and web pages.

Here are some hints and tips based on the Signum Technical Support engineers' experience with JTAG-related issues on a wide variety of target boards.

- Always power the JTAGjet emulator first by attaching a USB cable to it; power-up the target board afterwards. This allows the JTAGjet to take control of all the JTAG lines right from the power-up moment and prevents placing an unnecessary load on the target.

Board Design Considerations

- Place the JTAG connector close to the CPU.
- The PD/VTRef power pin on the JTAG connector should be connected to the CPU I/O voltage. The JTAGjet emulator will drive all output lines with the same voltage. Do not apply 5V to the PD/VTRef pin, if the CPU I/O voltage is 3.3V or 1.8V.
- Do not use any serial resistors (33 Ohm or similar) on the PD/VTRef line on JTAG. The buffers on the JTAG probe are powered through this line, while and serial resistor may create an undesirable voltage drop.
- Connect all GND lines on the JTAG connector to ground on your board.
- You can leave DBGRQ and DBGACK lines unconnected. In newer designs, these pins are not used.
- Avoid too strong (2K or smaller) pull-up and pull-down resistors. In most cases, 10K is appropriate.
- Try to avoid level translation or other additional buffers on the target JTAG lines. Use the correct type of the JTAGjet LV (low voltage) probe instead.
- Do not route the JTAG signals to many connectors on the board – use passive adapters outside of the board.

- Do not short-circuit the nSRST and nTRST lines. Otherwise, you may not be able to stop the CPU at the reset vector since the nTRST line resets the JTAG logic.
- As the system reset line (nSRST) is bi-directional, connect it directly to the CPU pin rather than to the RESET switch on the board before the reset circuit. This will allow the debugger to monitor the status of the RESET button on the board and detect the moment the CPU is released from reset.
- Do not tie hard the EMU0/EMU1 lines on the 14-pin TI-style connector to VCC or GND. These bidirectional lines control the CPU boot mode and are driven high by the JTAGjet. Provide appropriate pull-up and pull-down resistors on the board as needed.
- Certain processors have JTAG lines available on more then one pin assigned during power-up. Make sure that the correct CPU boot mode has been selected.
- If your target drives large currents, as often is the case with the motor-control C2000 DSP. Use the JTAG isolator probe ADA-ISO-TI14 to minimize the risk of emulator damage due to ground loops and excessive noise on the GND and power lines.
- Connect the RTCK pin on the JTAG connector to the RTCK pin on the device, if the latter is available. It will allow the emulator to work with Adaptive/RTCK clock. The JTAG clock will adapt to the changing CPU frequency.
- Do not use fixed clocks if the processor, such as the ARM7TDMI-S, features a synchronized ARM core. A CPU application slowing down the clock or going to sleep may cause debugging problems.
- Do not use the Adaptive clock on boards with the lines TCK and RTCK shorted, as often occurs on the TMS470 and DSP boards. Use a fixed clock instead.
- Avoid using fast clocks when connecting to a new board for the first time.

- Use clocks faster than 10MHz only if you are confident that the board and the device will allow fast JTAG clocks.
- If you are connecting the JTAGjet emulator to a CPU simulated in FPGA or to another hardware simulator, select a slower JTAG clock. Processors simulated in FPGAs tend to run slow.
- In multi-processor systems, one processor is often designated as a master that may hold the other processors in reset. Make sure that such a master CPU releases the slave processors from reset when debugging one or more of the slave processors.

Updating the Emulator Firmware

EmuDiag allows you to upgrade the firmware in your USB emulator after a recommendation to do so from Signum Systems Technical Support team. Before updating the firmware, ensure that:

- The emulator is connected only to the USB port; it should not be connected to a target board.
- The manufacturer-provided power supply is used, especially with JTAGjet-ETM model.
- The connection to an emulator is stable (verify it by using the Test Connection feature several times).
- You are not running any CPU-intensive or video display applications on your computer to minimize the chances of a hang-up during the update.

Click the Update Firmware button and navigate to the .upg firmware image file that is applicable to your emulator and select it (Figure 29).

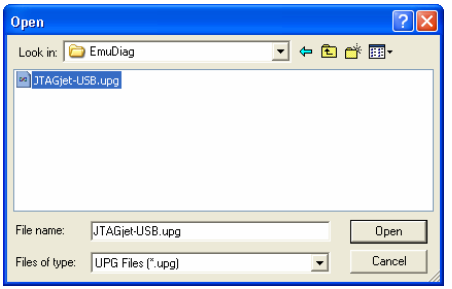


FIGURE 29 Selecting a firmware image file during a firmware update

The .upg emulator upgrade files are not part of the EmuDiag installation. They are delivered from Signum to the user via email. Uncompressing the upgrade package into the same folder where EmuDiag.exe is located will facilitate file browsing.

EmuDiag starts loading the new firmware in steps. The updated process lasts 30 to 60 seconds. Do not disconnect the emulator and do not close EmuDiag before the process ends. An update log of an update is shown in Figure 30.

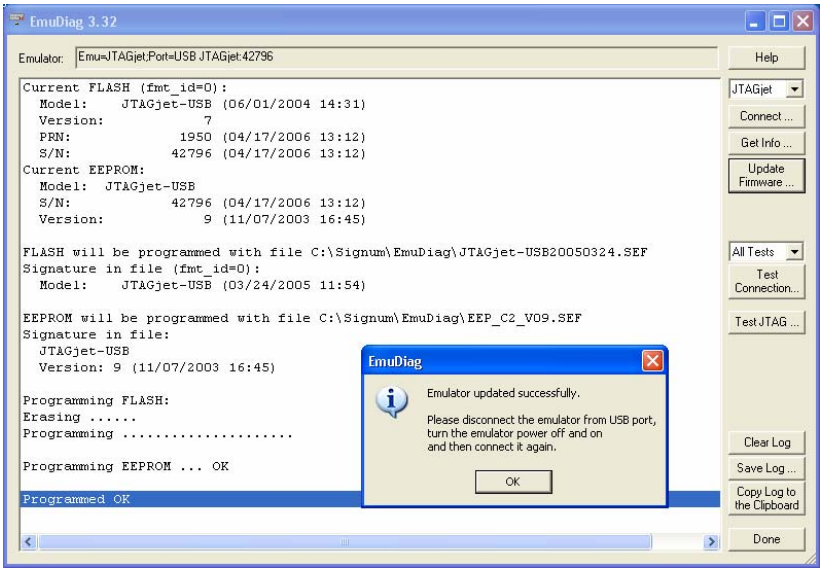


FIGURE 30 Upgrading the JTAGjet firmaware ver. 8 to ver. 9.

To make verify that the newly updated firmware runs, turn the emulator power off by disconnecting the USB cable when prompted. Reattach the USB cable and click the Connect button to re-connect. The updated firmware version—9 in our example—is displayed (Figure 31).

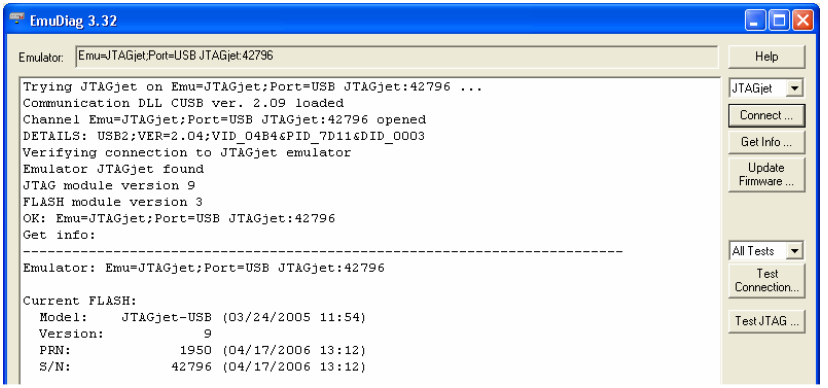


FIGURE 31 Reconnecting the emulator to verify the presence of the updated firmware.

Operations on Log Window

The *Clear Log* button erases the contents of the EmuDiag main window, eliminating the clutter that may result from invoking various EmuDiag functions.

The *Save Log* button saves the contents of the EmuDiag main window in a text file. The location and name of the file can be selected using the file browser brought up after this button is pressed. In addition to this log file, EmuDiag automatically creates in its folder a few other diagnostic files (with the .log extension) on as needed basis. These files provide accurate information about your emulator and its behavior, and therefore should all be sent to Signum Technical Support if you suspect that the emulator malfunctions.

The *Copy Log to Clipboard* button duplicates in the Windows clipboard the entire log of the current EmuDiag session. This log can be later pasted to email or any other document. This log is identical with the contents of the file saved using the Save Log button.

Testing a Malfunctioning Emultor

If your emulator does not seem to function properly, follow the procedure below to determine if the unit is indeed broken.

1. Download the latest **EmuDiag.zip** diagnostic utility from our ftp site at ftp://ftp.signum.com/pub/emudiag.zip.

2. After installation, run the **EmuDiag.exe** program with the emulator connected to a powered target board.

3. Click on **TEST JTAG** and click on **Test JTAG Clock** (JTAG clock should be set at 10MHz). If the test fails, test it at 1MHz. If the emulator still fails it needs repair.

4. Make sure that the JTAG Header shows **14 pin TI DSP/OMAP**. Click the **More** button.

5. Test the CPU Reset and TRST by pressing the **Detect** button. If the test fails, the emulator needs repair.

6. In the JTAG activity header picture set the checkbox **Read every 0.5 sec** and double click on **EMU0** pin. It should turn from black to white. Double click EMU0 again to make it black again. If this test fails, the emulator needs repair.

7. Do Step 6 on the EMU1 pin.

❑