FPSLIC™ System Designer™ 3.0

User Guide

Table of Contents

Section 1

Introduction

Atmel's System Designer™ allows designers to create fast and predictable designs with AT94K Field Programmable System Level Integrated Circuit (FPSLIC™) and AT94S Secure FPSLIC devices.

System Designer is a tool developed by Atmel specifically for Field Programmable System Level Integrated Circuit (FPSLIC) devices. It includes ModelSim®, LeonardoSpectrum™ and Co-verification powered by Mentor Graphics®.

Available for use with Windows® 95/98/2000/Me/XP and WindowsNT®-based computers, System Designer combines industry-standard software for design entry, synthesis and simulation with Atmel's proprietary software for component generation, automatic and interactive placement and routing, timing analysis and bit stream generation.

What's New in System Designer 3.0

- Support
  - Windows XP
  - AT94S Secure FPSLIC
  - HDLPlanner™ for Editing and Creating HDL Designs (VHDL®/Verilog®)
  - EasyPlanner™ for Organizing Designs and Library Components
- Functionality
  - Microsoft® Windows Native File Browsers
  - New Attach Files for Reporting a Problem
- New User Interface
  - Quick Flow, Advanced Flow and Device Views
  - Device Programming in the Advanced Flow View
  - Project Setup
  - Online Help
  - Log Viewer's Toolbar Highlights Errors and Warnings

- Figaro IDS New Features
  - I/O Pad Attribute Editor: Set Defaults option to apply same value for all rows
  - Constraints Editor: Import File option for `*.rct` files
  - Options > Place And Route: autoSetParameters enables or disables the scroll bar
  - Assign Pin Locks Dialog: Import File option for `*.pin` files. Save File saves the `*.pin` file using alphabetical order
  - New Batch Commands
  - HDL Planner: New wizard for creating new files, Macro Map Dialog offers control over the instantiation parameters, syntax highlighting
- Co-Verification
  - Modelsim 5.6e
  - AVR Studio® 4.07
  - FPSLIC JTAG ICE Support for On-Chip Debugging from AVR Studio
  - Single button Co-verification Restart at Line 0
  - Hardware and Software Status
  - Auto Hardware Break Capability after the Completion of any Stepping in AVR Studio
  - Capability to Launch AVR Studio on a Selected AVR Studio Project or an Existing Object File from the Command Line
- AVR Studio 4.07 Features
  - Modular Design for Easy and Flexible Upgrade Paths and Third Party Software Integration with Plug-ins and AVR Studio API Management
  - New User Interface with Docking Windows andToolbars
  - Project Management for Assembler with Syntax Coloring Editor
  - Powerful High-level Source Debugging: Unlimited Conditional Program Breakpoints, Run, Single-step, Step-over, Step-out, Set Current Statement and Run to Cursor
  - Watch, I/O Register View, Memory View, Processor View, Disassembly View, Breakpoints View
  - Supported Languages: C/C++, Assembler and Pascal

Section 2

Installing System Designer

2.1 System Requirements

For a single-user system, System Designer requires the following:

Hard Disk	250-Mbyte Minimum
Operating System	Windows 95/98/2000/Me/XP, or WindowsNT 4.0 with Service Pack 6
RAM	128-Mbyte
Peripherals	Parallel Interface Port
Network	Network Interface Card or Security Dongle
Browser	Internet Explorer 5 or above
Privileges	Administrative Privileges

The software security dongle is used to generate a unique HOSTID for systems without a network interface card. The security dongle is connected to the PC through the parallel port interface. It is possible to configure a floating network license through the security dongle. The security dongle allows users to use the software dongle on different machines by removing and placing the dongle on other machines.

2.2 System Designer Installation

1. Insert the supplied System Designer CD-ROM into the computer. If the CD does not automatically start, execute `SETUP.EXE` from the CD.
2. From the CD Browser, select *Install Products* and select *System Designer*, see Figure 2-1.

Figure 2-1. Install Products

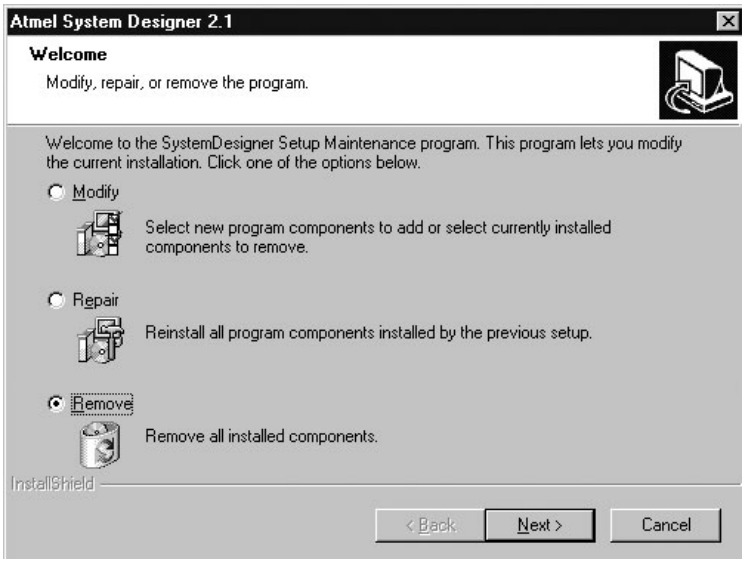
If you have a previous version of System Designer installed on your machine, the installation wizard will detect it, see Figure 2-2.

Figure 2-2. Detection of a Previous Version of System Designer

3. Press *Yes* to remove the previous version of System Designer and continue with the installation. *No* will cancel the installation.

   InstallShield® will guide you through the setup. The *Modify, repair or remove the program* dialog box appears, see Figure 2-3.

Figure 2-3. Modify, Repair or Remove the Program Dialog Box

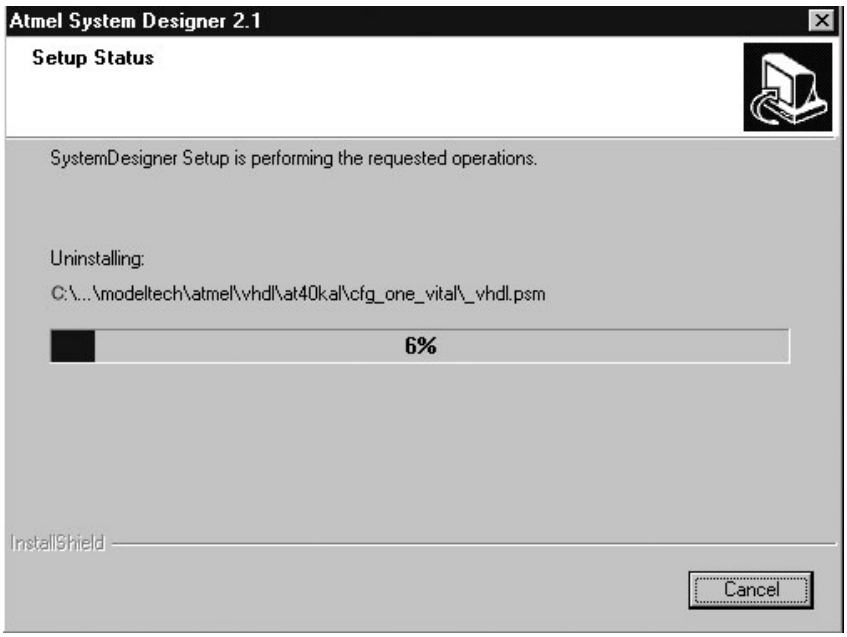


4. Select *Remove* and press *Next >*. The *Confirm File Deletion* dialog box appears, see Figure 2-4.

Figure 2-4. Confirm File Deletion Dialog Box

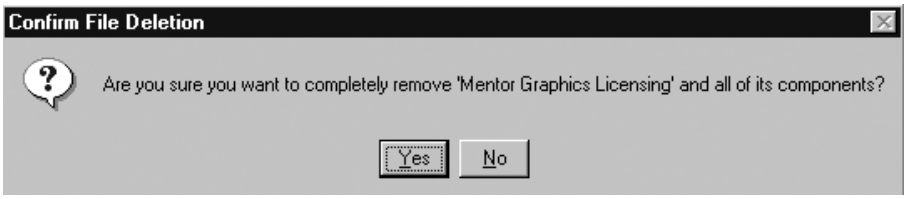
5. Press *OK*. The System Designer setup will remove the current version, see Figure 2-5.

Figure 2-5. Setup Status



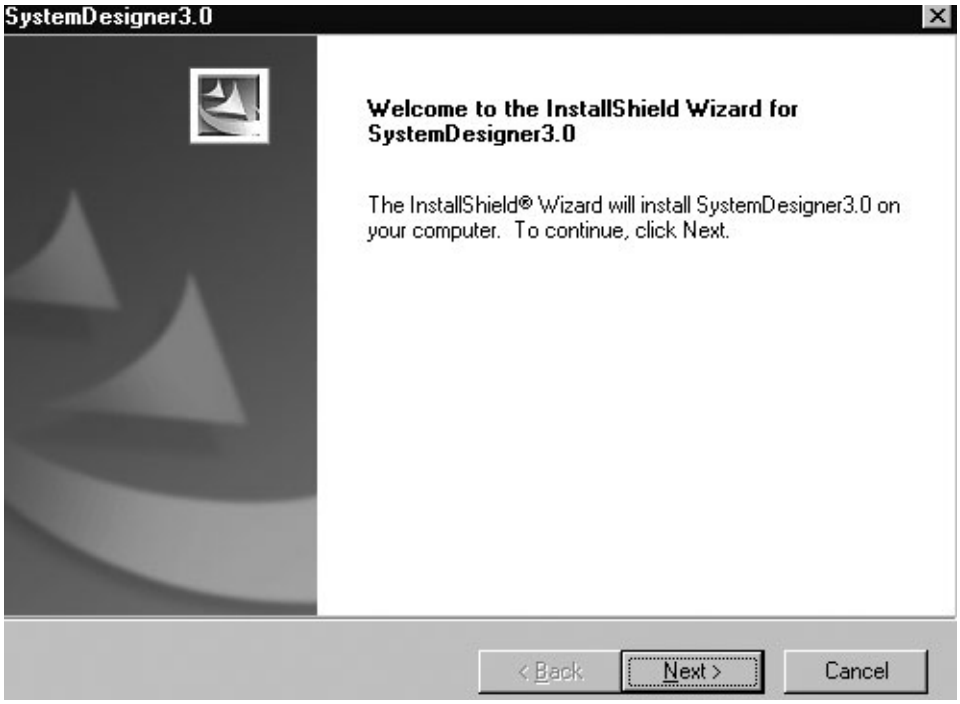
InstallShield will then remove Mentor Graphics Licensing. The *Confirm File Deletion* dialog box appears, see Figure 2-6.

Figure 2-6. Confirm File Deletion Dialog Box



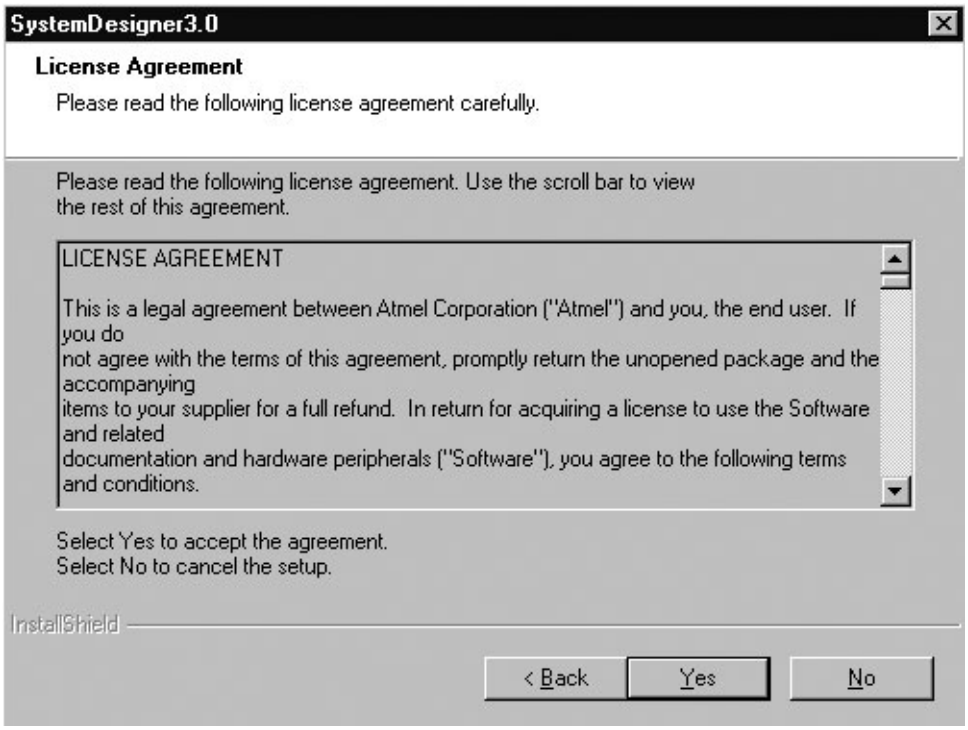
6. Press *Yes*. The System Designer 3.0 installation dialog appears, see Figure 2-7.

Figure 2-7. .System Designer 3.0 Installation Dialog



7. Press *Next >*. The License Agreement dialog appears, see Figure 2-8.

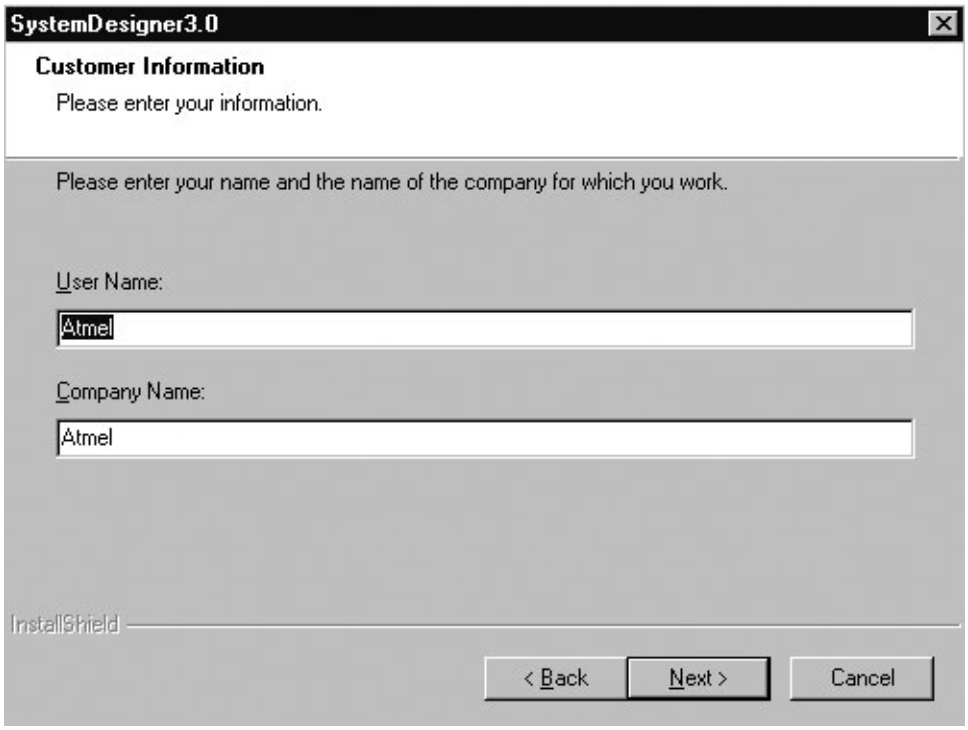
Figure 2-8. License Agreement Dialog



8. Read the License Agreement and press *Yes*. You must accept this agreement if you want to install System Designer. If you choose *No*, the setup will close.

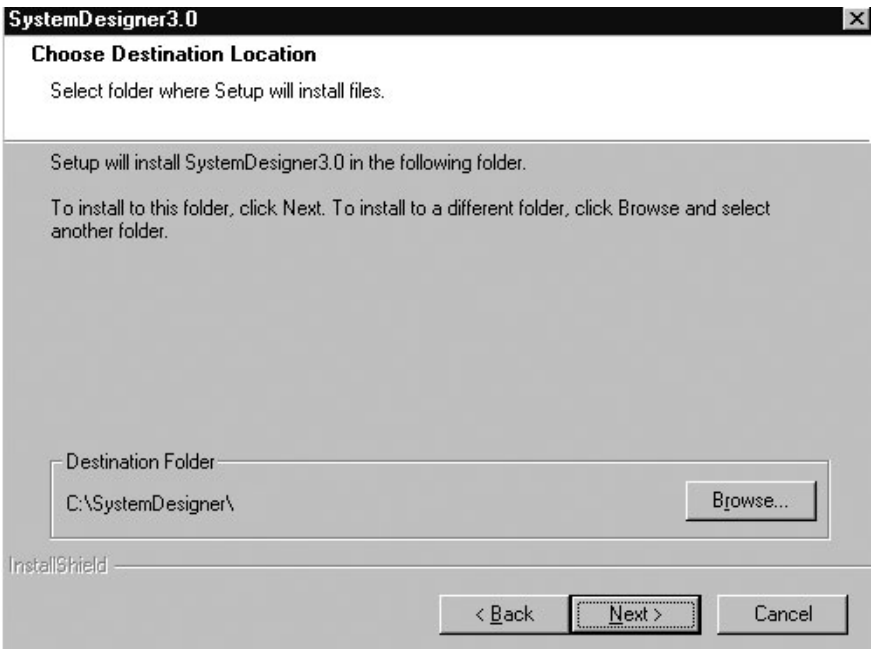
   The *Customer Information* dialog box appears, see Figure 2-9.

Figure 2-9. Customer Information Dialog Box



9. Enter the requested information and press *Next >*. The *Choose Destination Location* dialog box appears, see Figure 2-10.

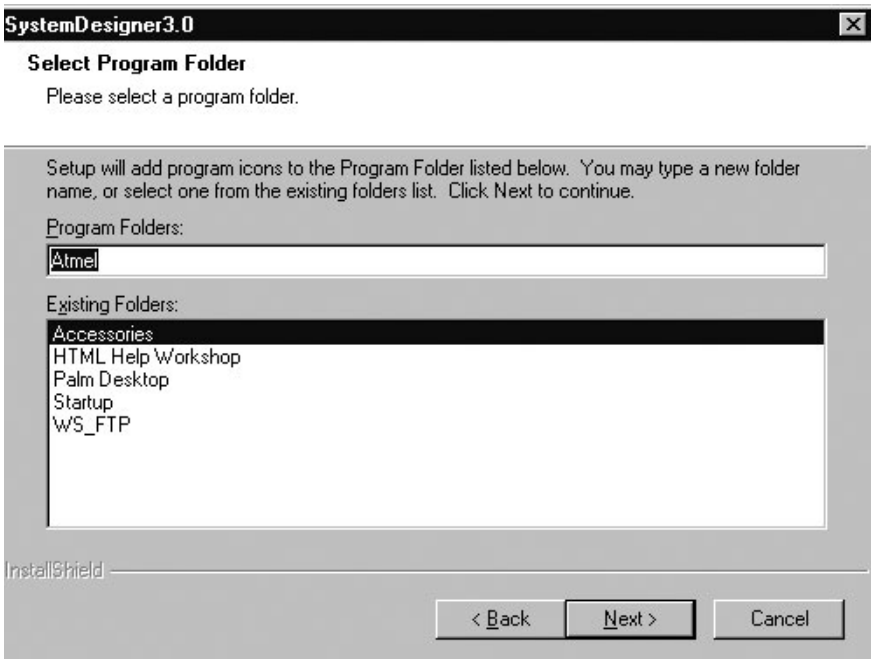
Figure 2-10. Choose Destination Location Dialog Box



System Designer's default installation path is `C:\SystemDesigner`. If you prefer to use another path, press the *Browse* button to navigate to the destination folder. Do not use spaces between words.

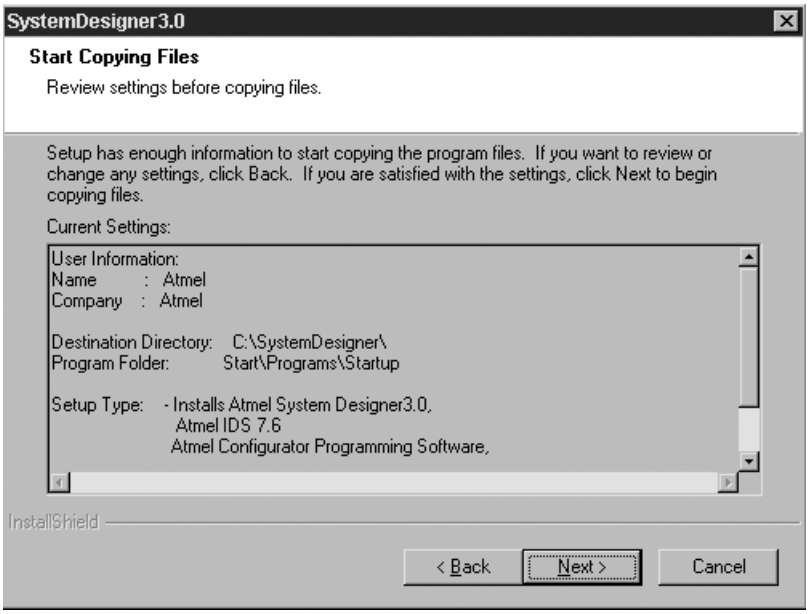
10. Press *Next >*. The *Select Program Folder* dialog box appears, see Figure 2-11.

Figure 2-11. Select Program Folder Dialog Box



11. Take the default and press *Next >*. The *Start Copying Files* dialog box appears, see Figure 2-12.

Figure 2-12. Start Copying Files Dialog Box

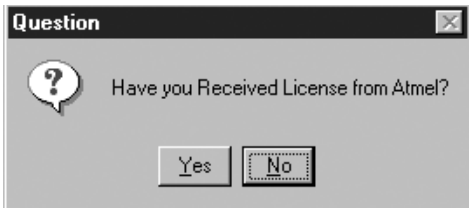


12. Review the current settings and press *Next >* to proceed with the installation.
13. The installation will require to restart your computer, see Figure 2-13. Press OK.

Figure 2-13. Restart Window

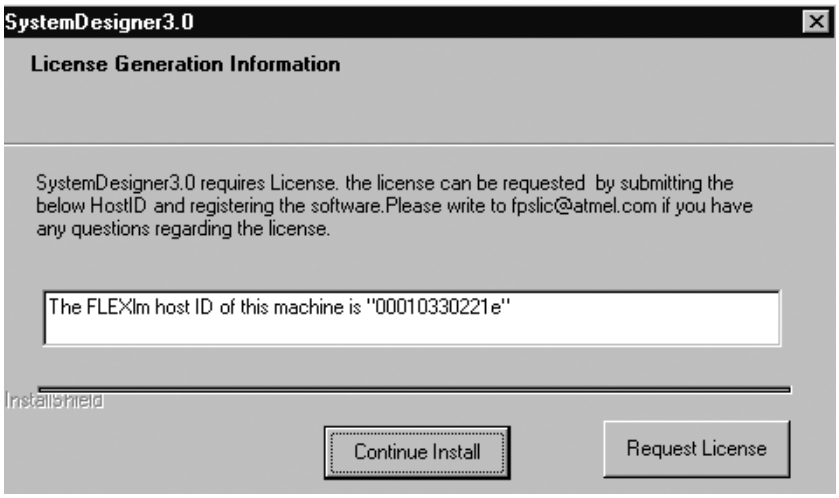
Once your computer has been restarted, a dialog box asking if you have received a license from Atmel appears, see Figure 2-14.

Figure 2-14. Information Dialog Box - Atmel's License



– Press *Yes* if you have received a license.
– Press *No* if you don't have one. A dialog box with your Host ID appears, see Figure 2-15.

Figure 2-15. License Generation Information

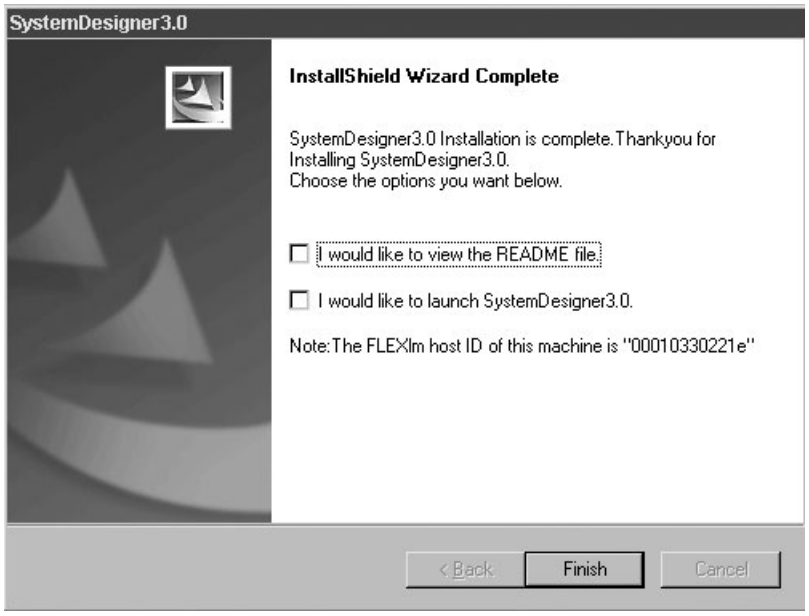


– Press *Request License* to request a license now. The *FPSLIC Licensing* page appears, (www.atmel.com/products/fpslic/forms/sublicense.asp). Fill in the form and press the *Submit* button. You will receive a license by email within the next 24 hours.

– Press *Continue Install* if you want to request your license later.

14. A dialog box asking you view the README file or to launch System Designer appears, see Figure 2-16.

Figure 2-16. InstallShield Wizard Complete Dialog



15. Select your option and press *Finish*.

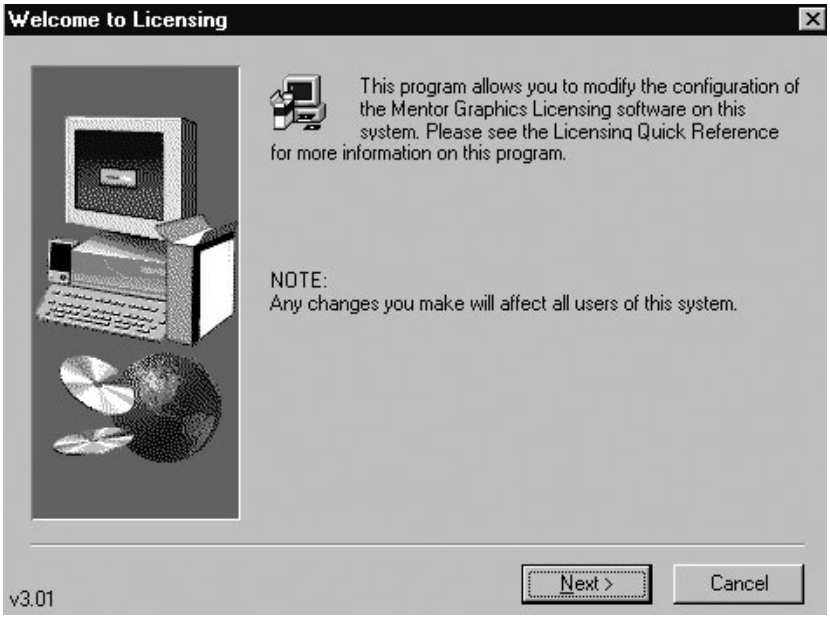
2.3 Configuring the Product License

If you are using a dongle-based license, refer to “Installing a Dongle-based License” on page 11.

You must have a valid license in order to proceed.

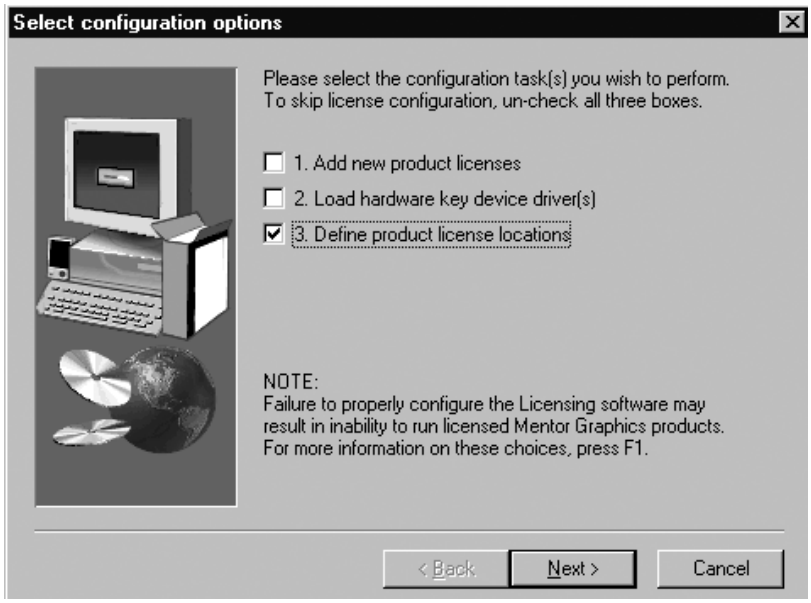
1. Save your license under `C:\SystemDesigner\fpslic.dat`.
2. Go to the *Start* menu and choose *Programs > Atmel > Mentor Graphics Licensing > Configure Licensing*. The *Welcome to Licensing* dialog box appears, see Figure 2-17.

Figure 2-17. Welcome to Licensing Dialog Box



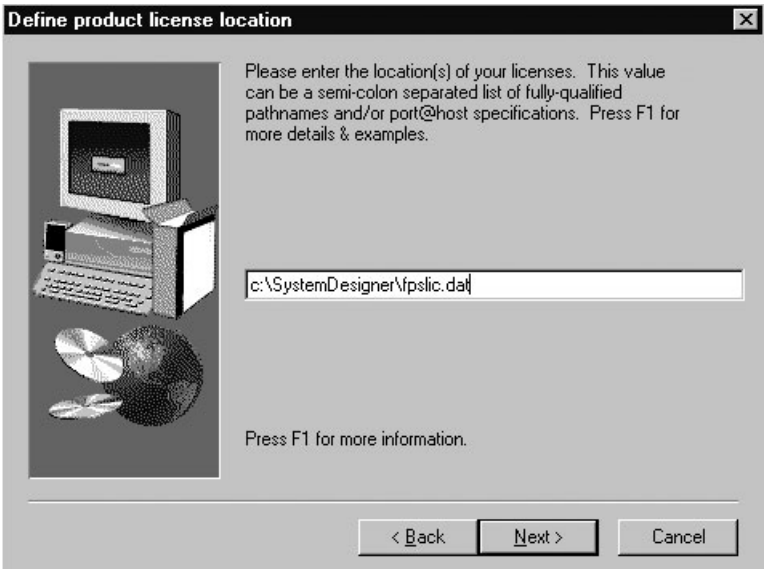
3. Press *Next >*. The *Select Configuration Options* dialog appears, see Figure 2-18.

Figure 2-18. Select Configuration Options Dialog Box



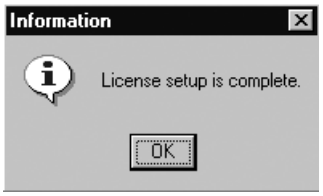
4. Select *Define Product License Locations* and press *Next >*. The *Define Product License Locations* dialog box appears, see Figure 2-19.

Figure 2-19. Define Product License Location Dialog Box



5. Specify the path `C:\SystemDesigner\fpslic.dat`. Press *Next*. An *Information* dialog box appears, see Figure 2-20.

Figure 2-20. Information Dialog Box – License Setup Complete



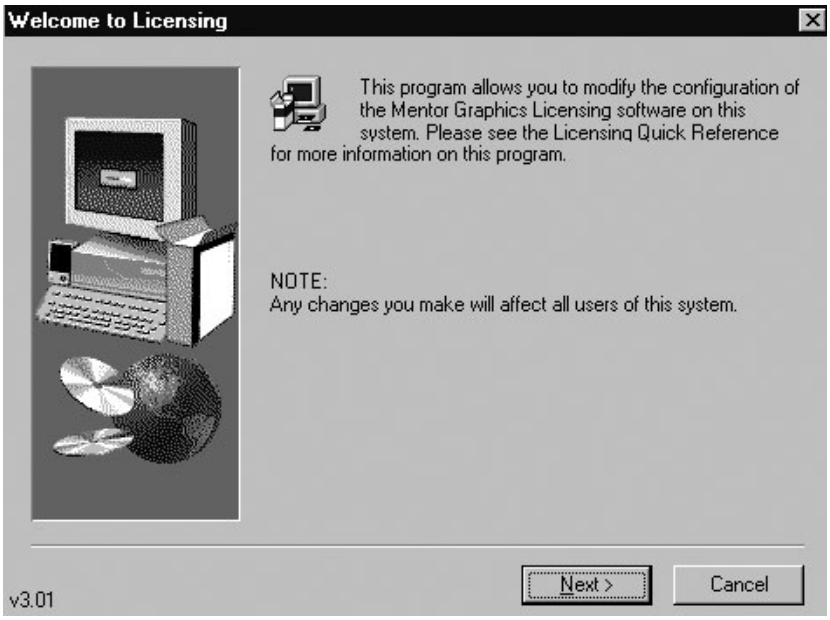
6. The license setup is now complete, press *OK*.

2.4 Installing a Dongle-based License

Follow the steps below if you are configuring a dongle-based license. You must have an active license to proceed.

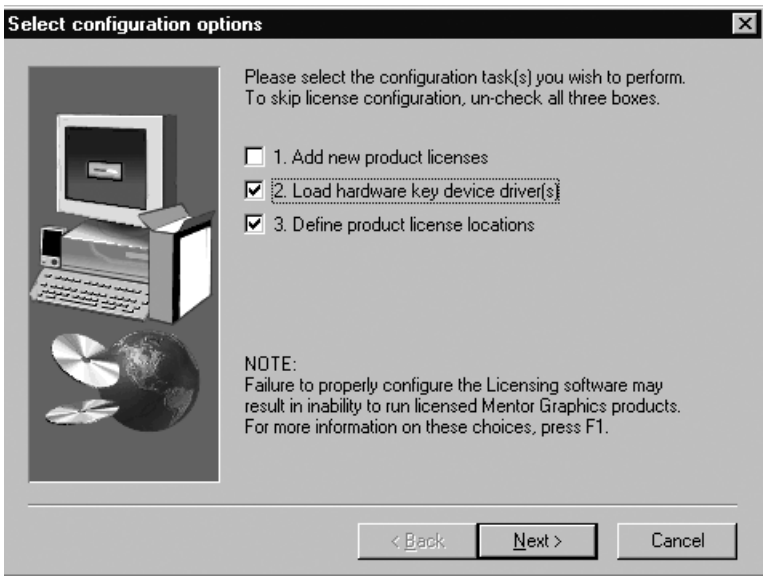
1. Save your license under `C:\SystemDesigner\fpslic.dat`.
2. Go to the *Start* menu and choose *Programs > Atmel > Mentor Graphics Licensing > Configure Licensing*. The *Welcome to Licensing* dialog box appears, see Figure 2-21.

Figure 2-21. Welcome to Licensing Dialog Box



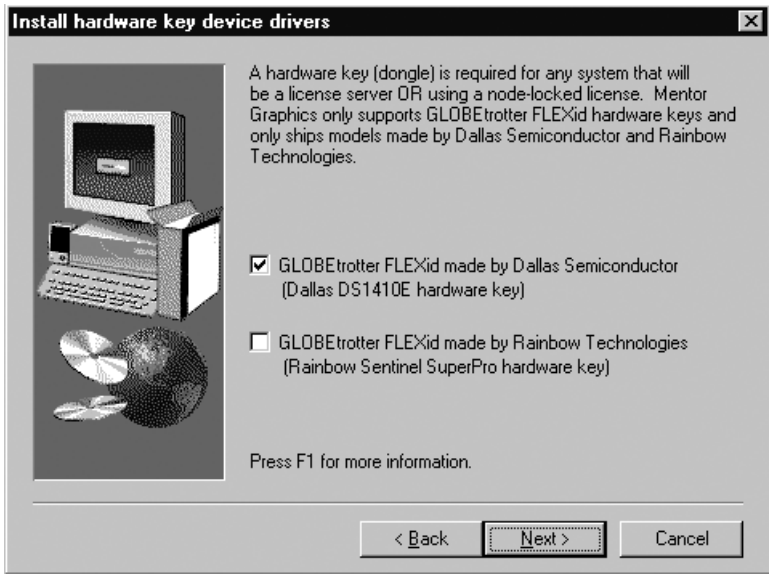
3. Press *Next >*. The *Select Configuration Options* dialog box appears, see Figure 2-22.

Figure 2-22. Select Configuration Options Dialog Box



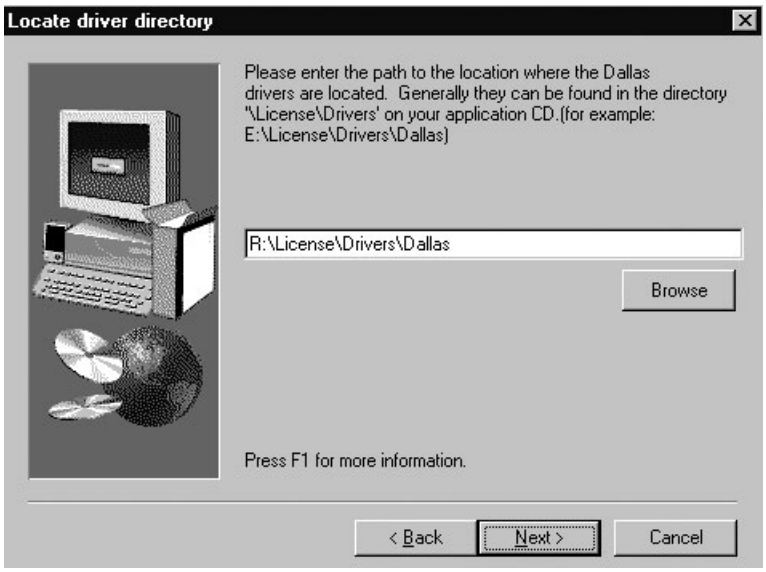
4. On the *Select Configuration Options* dialog box select *Load Hardware Key Device Driver(s)* and *Define Product License Locations*. Press *Next >*. The *Install Hardware Key Device Drivers* dialog box appears, see Figure 2-23.

Figure 2-23. Install Hardware Key Device Drivers Dialog Box



5. Select *GLOBEtroter FLEXid made by Dallas Semiconductor (Dallas DS1410E hardware key)* and press *Next >*.The *Locate Driver Directory* dialog box appears, see Figure 2-24.

Figure 2-24. Locate Driver Directory Dialog Box

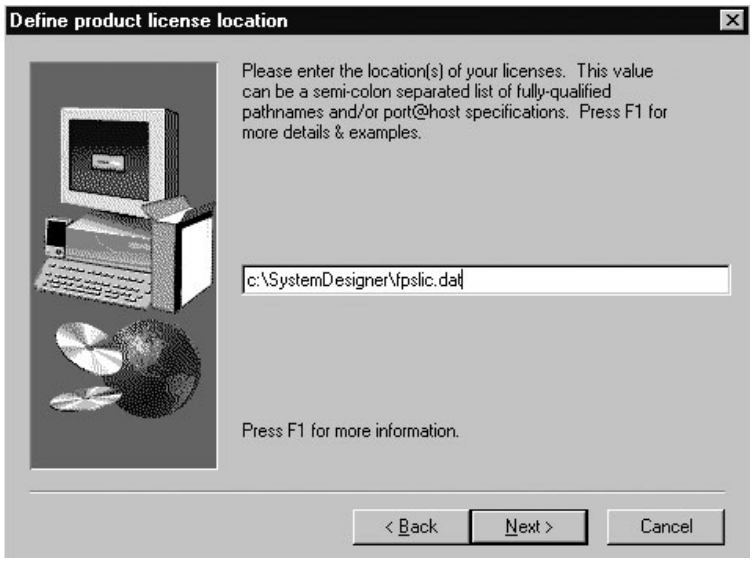


If the System Designer CD is placed in the CD-ROM drive, it will automatically identify the path where the driver information is located. If this is not the case, use the *Browse* button to locate `<CD Drive>:\License\drivers\Dallas`. Press *Next >*, an *Information* dialog box appears, see Figure 2-25.

Figure 2-25. Information Dialog Box – Restart Windows

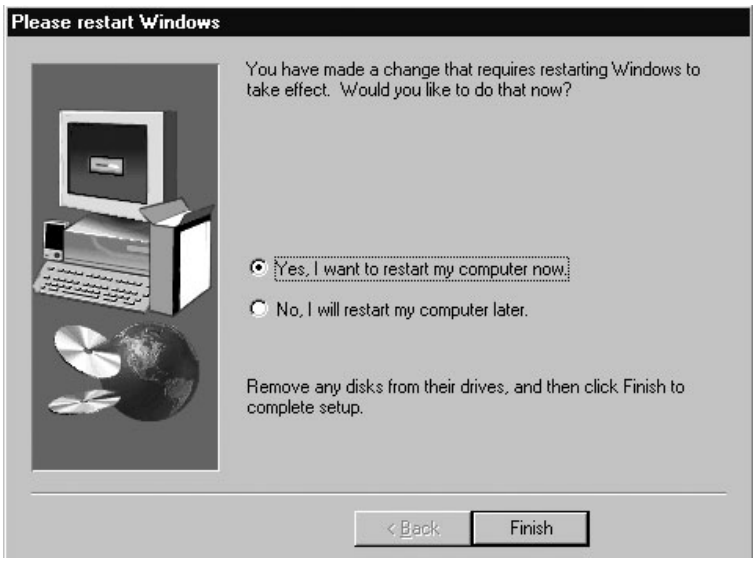
6. Press *OK*. The *Define Product License Location* dialog box appears, see Figure 2-26.

Figure 2-26. Define Product License Location Dialog Box



7. Type `C:\SystemDesigner\fpslic.dat` and press *Next >*. The *Please Restart Windows* dialog box appears, see Figure 2-27.

Figure 2-27. Please Restart Windows Dialog Box



8. Remove any disks from their drives and press *Finish* for the licensing to take effect. The license setup is complete.

2.5 License Troubleshooting

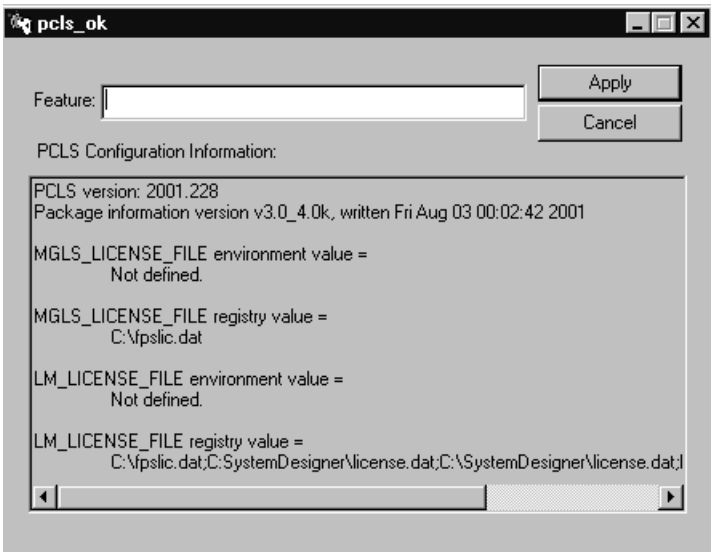
In order to check if all the licenses work, follow the procedure below:

1. Open your `fpslic.dat` file, the file should contain the 4 product names as shown below:

```
INCREMRENT cveatmel1...
INCREMRENT atmelmti...
INCREMRENT leospecls1...
INCREMRENT leospecls1atmel...
```

2. Go to the *Start* menu and choose *Programs > Atmel > Mentor Graphics Licensing > pcls_ok*. The *pcls_ok* dialog box appears, see Figure 2-28.

Figure 2-28. pcls_ok Dialog Box



3. Type the first *Feature* name (`cveatemel1`) and press *Apply.* If the license was installed successfully, the *PCLS_OK* dialog box appears, see Figure 2-29.

Figure 2-29. PCLS_OK Dialog Box

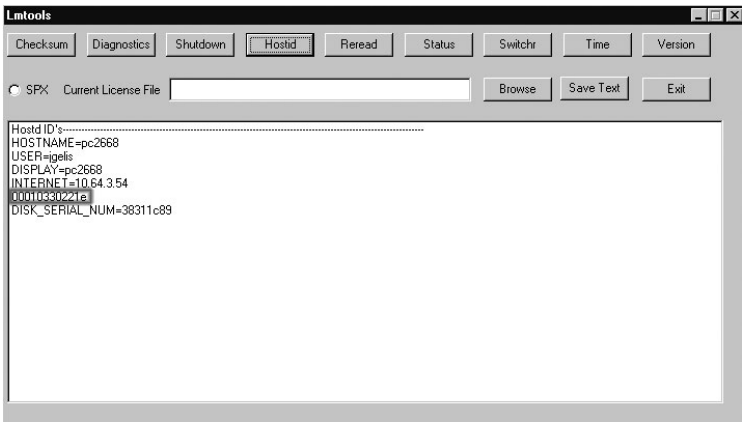
4. Repeat the same procedure with the rest of the products (`atmelmti`, `leospecls1`,`leospecls1atmel`).

If after checking all the licenses you still have problems launching Leonardo, check for an expired FPSLIC license file that may be located in `c:\flexlm` directory. If found remove the expired license and try launching Leonardo again.

If both ModelSim and Leonardo fail to launch, check if the host ID matches with your license file.

5. Go to the *Start* menu and choose *Programs > Atmel > Mentor Graphics Licensing > Lmtools*. The *Lmtools* window appears.
6. Click on *Hostid* and write down the number, see Figure 2-30.

Figure 2-30. Lmtools Window



7. Open the `fpslic.dat` file and compare the host ID. If it is the same host id, check one of the other options below. If it is different, go to the *FPSLIC Subscription and Licensing* and submit your request for a new license.
8. Check for the expiration date on the license file.
9. Check if the path in your `autoexec.bat` file matches as shown below:

```
SET
PATH%PATH%;c:\SystemDesigner\bin;c:\SystemDesigner\Mentorgraphics\cve_home.ixn\bin;c:\SystemDesigner\Mentorgraphics\cve_home.ixn\lib
SET ATMELDIR=c:\SystemDesigner\etc
SET FIGARO_HOME=c:\SystemDesigner
SET CVE_HOME=c:\SystemDesigner\MentorGraphics\CVE_HOME.IXN
SET MGLS_HOME=c:\SystemDesigner\MentorGraphics\CVE_HOME.IXN\MGLS
SET MGC_CVE_MAX_SHMEM_SIZE=3
SET PCLS_DIR=c:\Mentor~1\Licens~1
SET PATH=%PATH%;%PCLS_DIR%
```

Section 3

FPSLIC JTAG In-Circuit Emulator

The JTAG In-Circuit Emulator is a 4-wire Test Access Port (TAP) controller compliant with the IEEE 1149.1 standard. The JTAG ICE is a complete tool for performing On-Chip Debugging on all AVR 8-bit RISC microcontrollers and all FPSLIC devices that support the JTAG Interface, see Figure 3-1. This document focuses on the On-Chip-Debugging (OCD) procedure on all the FPSLIC devices.

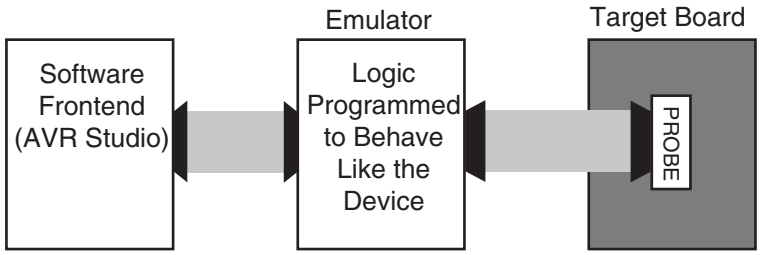
The FPSLIC JTAG protocol gives the user the capability to view and control the internal resources of the FPSLIC device and perform real time emulation while the FPSLIC device is running on the target system. The FPSLIC JTAG ICE is supported only by AVR Studio 4.07 and not by any previous versions, System Designer populates AVR Studio with the FPSLIC specific files and makes it capable to emulate or simulate the FPSLIC devices.

Figure 3-1. The JTAG ICE

3.1 Emulation vs. On-Chip Debugging

A traditional emulator is a device built exclusively to imitate the behavior of the device that it is meant to emulate. During emulation, the user connects the emulator to the target board instead of the actual device, see Figure 3-2.

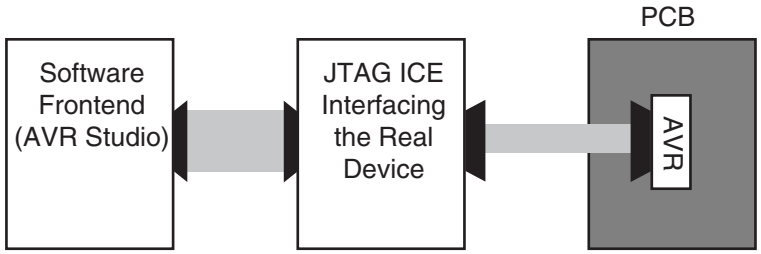
Figure 3-2. Traditional Emulators Logic



However, the JTAG emulator does not replace the actual device. While the FPSLIC device is normally connected to the target board, the JTAG emulator is connected to the On-Chip Debugging interface in the actual device, allowing to control and monitor the action that is taking place in the device through the JTAG (IEEE 1149.1 compliant) interface.

All FPSLIC devices with a JTAG compliant interface are provided with an OCD logic that interfaces with the JTAG emulator. This logic is capable of controlling the code execution in the FPSLIC device. So while a traditional emulator emulates device behavior, the JTAG ICE takes control of the device and executes the code in a physical device, see Figure 3-3.

Figure 3-3. The OCD Logic



The OCD system provides exact electrical and timing characteristics, but a traditional emulator will normally have better visibility and control of the internal resources of the device. Features like trace buffer are not possible using the FPSLIC OCD system. Since the OCD system is built into the device, some restrictions may apply, refer to “Known Issues” on section 3.4 for more information.

The following sections explain the differences between JTAG ICE and a traditional ICE operation while debugging code.

3.1.1 Run Mode

In Run mode the code execution is completely independent of the JTAG ICE. The JTAG ICE will continuously poll the target FPSLIC device to see if a break condition has occurred. When this happens the OCD system will read out all necessary data (Program Counter (PC), I/O registers, general-purpose registers, SRAM contents and FPGA design logic signals in the FPGA-SRAM and FPGA-AVR boundaries) and transmit it to AVR Studio through the JTAG interface. Since the target FPSLIC device operates independently, this operation can only be checked until the first breakpoint.

3.1.2 Stopped Mode

When a breakpoint is reached, the program execution is halted, but all I/Os will continue to run as if no breakpoint occurred. As an example assume that a breakpoint is reached during a UART transmission: the transmission will be completed and corresponding bits will be set. The TXC (transmit complete) flag will be set and will be available on the next single step of the code even though it normally would happen later in an actual device.

3.1.3 Breakpoints

The FPSLIC OCD system distinguishes between two types of breakpoints: Software and Hardware breakpoints.

- Software Breakpoints. A software breakpoint is a break instruction placed in the Flash memory. When this instruction is executed, it will break the program execution.To continue execution a *start* command has to be given from the OCD logic. AVR Studio physically rewrites the placed breakpoint on an instruction as a break instruction in the Flash memory. When reaching this instruction the operation is halted. When starting the execution, the pending instruction is executed before continuing to execute instructions from the Flash memory.
- Hardware Breakpoints. In the OCD logic there are 5 registers capable of storing one memory address each. The JTAG ICE reserves one of these registers for internal use. The 4 remaining registers can be combined in different ways to generate valid break conditions.

Since software breakpoints require reprogramming the entire page where the breakpoint is located, hardware breakpoints are strongly recommended.

3.1.4 I/O Registers Visibility

The JTAG ICE has limitations when it comes to I/O visibility. Visibility is the ability to view the contents in all I/O locations. When an FPSLIC device reaches a breakpoint, the contents of all I/O registers are read out and presented in AVR Studio. Due to the fact that reading alters the contents in some registers, these registers will not be read. (E.g. Reading UART data register will clear the TXC bit). See “Known Issues” on section 3.4 to check the complete list of registers that not are accessible through the JTAG ICE OCD system.

3.1.5 Single Stepping

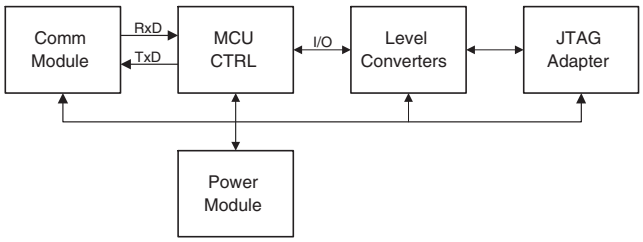
Some registers need to be read or written to within a specified number of cycles after a control signal is enabled. Since the I/O clock (and peripherals) continue to run at full speed in stopped mode, single stepping through such code will not meet the timing requirements. Between two single steps, the I/O clock might have run for millions of cycles. To successfully read or write registers with such timing requirements, the whole read or write sequence should be performed as an atomic operation running the device at full speed (i.e. use a macro, function call or run-to-cursor).

3.2 Hardware Description

Figure 3-4 shows the 5 main modules of the JTAG ICE.

- Communication Module
- Control Module
- Level Converter Module
- JTAG Adapter
- Power Supply Module

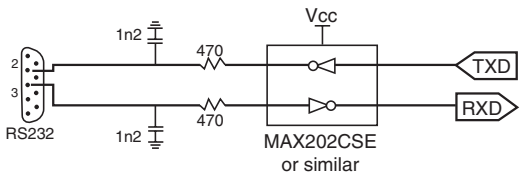
Figure 3-4. JTAG ICE Block Schematics



3.2.1 Communication Module

The JTAG ICE uses a standard RS-232 port for communication with AVR Studio. Figure 3-5 shows the block schematics for the communication module on the JTAG ICE.

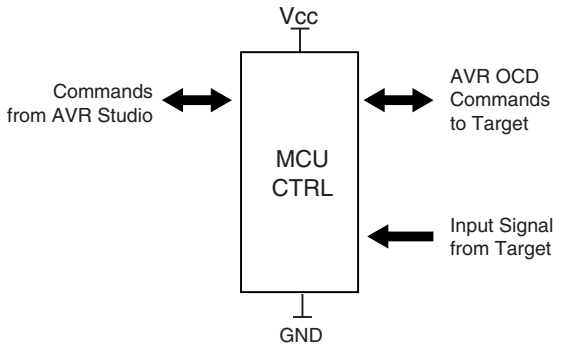
Figure 3-5. Communication Module Block Schematics



3.2.2 Control Module

The Control Module is responsible for converting the instructions from the front-end software to valid FPSLIC OCD instructions for the FPSLIC JTAG Interface. In addition it monitors and sets up a number of control signals. The block schematic for this module is shown in Figure 3-6.

Figure 3-6. Control Module Block Schematics



The JTAG ICE includes 3 status LEDs as shown in the table below:

LED	Description
Green	Indicates Target Power
Red	JTAG ICE Power indicator
Yellow	Indicates data communication over the JTAG interface

Note: When running from external power (9-15 Vdc), both red and green LEDs light.

3.2.3 Module Level Converters

The purpose of the Level Converters is to provide successful communication with target boards running at voltages different than the JTAG ICE itself. The target converter is designed to support target voltages from 1.8V to 6.0V.

The JTAG ICE can use an external power supply or take power directly from the target board. When voltage is taken from the target the valid voltage range is from 3.3V to 5.5V. For Power Supply details, please refer to “Power Supply” on section 3.2.4. The signals between the JTAG ICE board and the target board can be divided in three groups: output signals, input signals, and bi-directional signals.

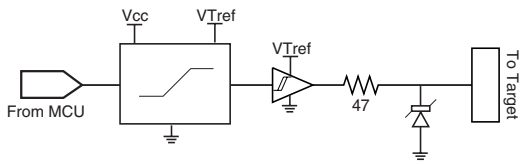
The JTAG ICE hardware supports more signals than are currently in use. This is done for compatibility with existing and future equipment. In this documentation only the signals routed to the JTAG adapter will be described. Signals in brackets [name] are implemented, but not used. Signal directions will be given as seen from the Control Module. E.g. TDI, which is the “Test Data Input” signal, will be described as an output since the direction of this signal is from the Control Module to the JTAG Device. A description of implemented and used signals can be found in the JTAG Adapter section.

The level converters use a buffer that limits the source and sink currents. The buffer is used in both outputs and inputs. The datasheet on the buffer 74HC2440, can be found at www.philips.com.

3.2.3.1 Output Signals

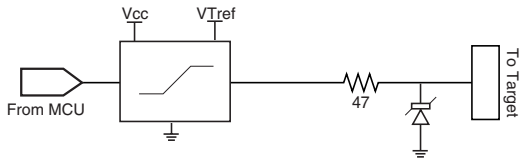
The JTAG ICE has implemented the following data output signals: TMS, TDI, TCK, [DBGRQ]. These signals are transmitted through the circuitry shown in Figure 3-7.

Figure 3-7. Level Converter Output



In addition, the nTRST output is implemented. Since this is a slow-changing signal, it is not sent through a pulse shaper line driver like the rest of the outputs. This output will act as an open collector output seen from the target, see Figure 3-8.

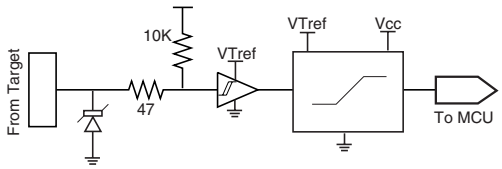
Figure 3-8. nTRST Open Collector Output



3.2.3.2 Input Signals

The JTAG ICE has implemented the following data input signals: TDO, [DBGACK, RTCK]. These signals are received through the circuitry shown in Figure 3-9.

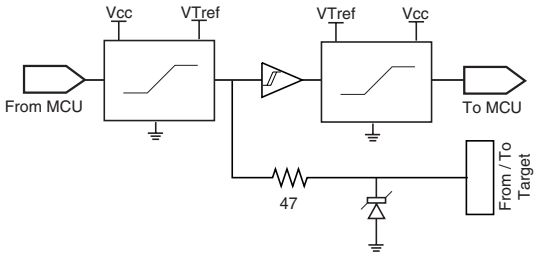
Figure 3-9. Level Converter Input



3.2.3.3 Bi-directional Signals

The only implemented bi-directional signal is the nSRST. This is an open collector output seen from the target. This pin is also an input so that the Control Module may observe a reset initiated on the target. The block diagram for the bi-directional I/Os is shown in Figure 3-10.

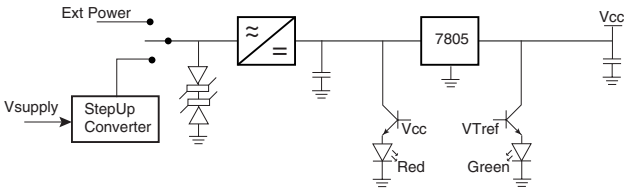
Figure 3-10. Level Converter I/O



3.2.4 Power Supply

The JTAG ICE power supply is implemented as shown in Figure 3-11. For details on considerations when drawing power from the target application, or using an external power supply refer to “Known Issues” on section 3.4.

Figure 3-11. Power Supply Block Diagram



3.2.5 JTAG Adapter

The JTAG adapter is shown in Figure 3-12. The 20-wire flex cable connects the JTAG adapter to the JTAG ICE. The JTAG adapter has two 10-pin connectors that have identical pin-out and signals. Use the one that best fits the target board. Only one of the connectors should be connected at any given time.

Figure 3-12. JTAG Adapter

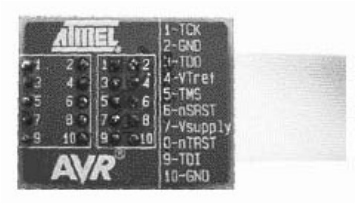
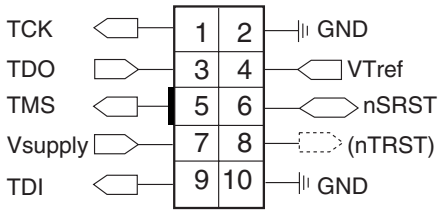


Figure 3-13 shows the JTAG connectors.

Figure 3-13. Close-up of the Connector



The signals are fully described in the JTAG Signals table below.

Pin	Signal	I/O	Description
1	TCK	Output	Test Clock, clock signal from JTAG ICE to target JTAG port
2	GND	–	Ground
3	TDO	Input	Test Data Output, data signal from target JTAG port to JTAG ICE
4	VTref	Input	Target reference voltage. VDD from target used to control logic-level converter and target power LED indicator
5	TMS	Output	Test Mode Select, mode select signal from JTAG ICE to target JTAG port
6	nSRST	Out-/In-put	Open Collector Output from Adapter to the Target System Reset. This pin is also an input to the adapter so that a reset initiated on the target may be reported to the JTAG ICE
7	Vsupply	Input	Supply Voltage to the Adapter, this connector can be used to supply the adapter with power from a regulated power supply (3 - 5) VDC (normally target VDD). This supply voltage input is automatically disconnected when a external power supply is connected
8	nTRST	NC(Output)	Not Connected, reserved for compatibility with other equipment (JTAG port reset)
9	TDI	Output	Test Data Input, data signal from JTAG ICE to target JTAG port
10	GND	–	Ground

3.3 On-Chip Debugging with FPSLIC JTAG ICE

The software of choice for performing emulation on the FPSLIC devices is AVR Studio. In order to use AVR Studio with FPSLIC devices, the System Designer Tool Suite has to be installed on your system.

This section will cover all the special features and considerations that the users need to know when On-Chip Debugging with AVR Studio and the FPSLIC JTAG ICE. Any topics regarding general use of AVR Studio are covered in the AVR Studio online help.

Before starting with the JTAG emulation, the user has to create and download a design to the target device. For more information on creating and downloading a design using System Designer, refer to the Quickstart tutorial,

Note: The FPSLIC OCD system must be enabled in the actual device: set the JTAG enable (SCR27) and the On-Chip Debug enable (SCR26) bits in the System Control Register (SCR). Also if the Configuration Memory Lockout bit is set (SCR4) the system will be in reset state and will not interact with the emulator.

3.3.1 Hardware Connections

Before starting AVR Studio, the JTAG ICE must be connected between the PC and the target board. All connections must be made before starting AVR Studio to ensure correct auto-detection by AVR Studio.

1. Connect the RS-232 cable between theJTAG ICE and a free COM port on the PC. AVR Studio will automatically search through the available COM ports and detect supported tools.

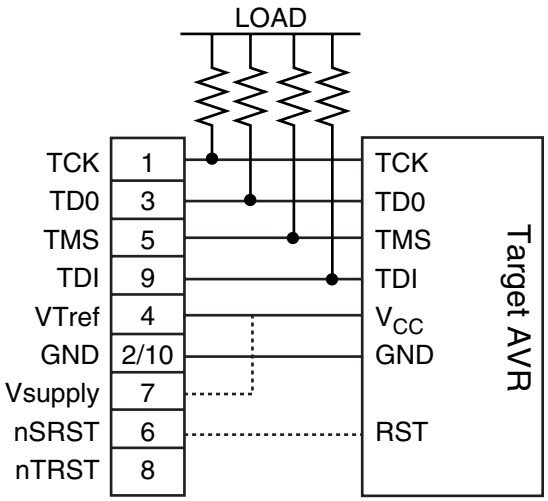
   Note: If there are other devices taking control over the COM ports, these have to be shut down before starting AVR Studio. AVR Studio cannot force control over a COM port, if other resources have control of the port (e.g. Modem, IrDA, PDA etc.)

2. Connect the JTAG cables to the TCK, TDO, TDI, TMS, VTref and GND pins of the target board. A minimum of 6 wires are required to connect the JTAG ICE to the target board. Vsupply and the nSRST are optional lines.

   Figure 3-14 shows which JTAG lines should be connected to the target AVR to ensure correct operation. To avoid drive contention on the lines, it is recommended that a series of resistors are placed between the JTAG lines and the external circuitry. The value of the resistors should be chosen so that the external circuitry and the AVR do not exceed their maximum ratings (i.e. sinks or sources too much current). See “Hardware Description” on section 3.2 for a detailed description of the hardware.

   Note: The JTAG ICE does not support several devices placed into a JTAG Chain. I.e. the target FPSLIC must be the only device connected to the JTAG ICE.

Figure 3-14. Connection of the JTAG ICE Cables to the FPSLIC Pins



Note: The nTRST signal is not used and is reserved for compatibility with other equipment. Vsupply should be used if the target board is supplying power to the JTAG ICE. nSRST is used to control and monitor the target reset line. This is however not necessary for correct emulation.

The ATSTK94 does not have a dedicated JTAG interface connector. To connect the JTAG ICE to the ATSTK94 board, the JTAG Probe must be strapped to the appropriate JTAG port pins of the target device. The FPSLIC JTAG pins will vary according to the FPSLIC device used. The following table summarizes the location of the FPSLIC TAP I/O ports on the all the available FPSLIC devices:

FPSLIC TAP Ports	AT94K05	AT94K10	AT94K40
TDI	IO34	IO50	IO98
TDO	IO38	IO54	IO102
TMS	IO43	IO63	IO123
TCK	IO44	IO64	IO124

The number of the pins in the following table corresponds to the FPSLIC Internal IOBs. The correspondent pins can be found on each device's reference manual. An example of the JTAG connections when using the ATSTK94 with an AT94KAL part is given in the following table.

JTAG Probe	ATSTK94
TDI	Pin 58
TDO	Pin 62
TMS	Pin 70
TCK	Pin 71

3. Connect the JTAG ICE to the power supply.

The JTAG ICE power supply is able to operate using both an external power supply or directly from the target board (the target application must be able to supply at least 220 mA @ 3.3V or 120 mA @5.5V to the JTAG ICE). The power connector works like a

switch when choosing between using external or target power. If no external power supply is connected, the board will use the target board as power source. If an external power supply is connected, this will be used instead of the target.

Voltage ranges for the external power supply are shown in the table below:

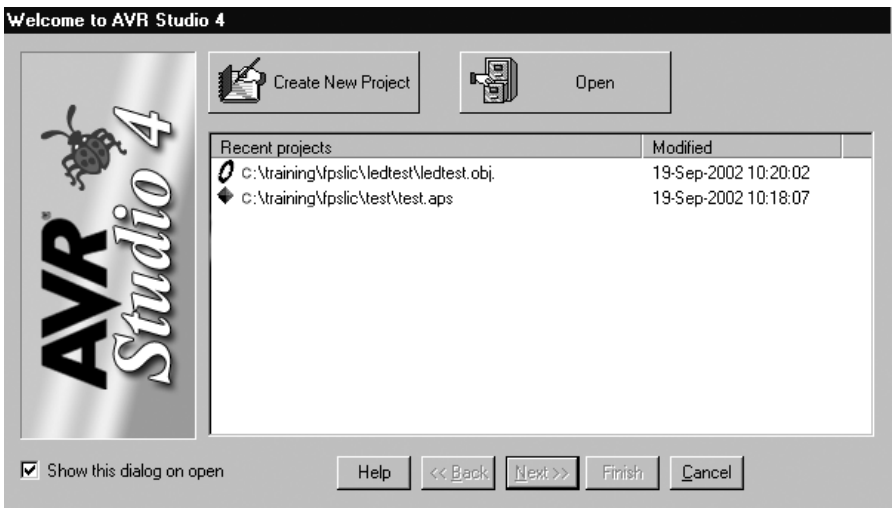
Power Requirements	Minimum	Maximum
Target VDD Voltage Range when Supplying the JTAG ICE	3.3V	5.5V
Total Target VDD Voltage Range	1.8V	6V
Target VDD Current Capability when Supplying the JTAG ICE	120 mA (@ 5.5V)	220 mA (@ 3.3V)
External Power Supply DC	9V (@ 50 mA)	15V (@ 60 mA)
External Power Supply Range AC	9V	9V

4. Power up the target board.
5. Power up the JTAG ICE. If the target board is supplying power to JTAG ICE, it will automatically be powered up correctly when powering up the target board.

3.3.2 On-Chip Debugging with FPSLIC JTAG ICE

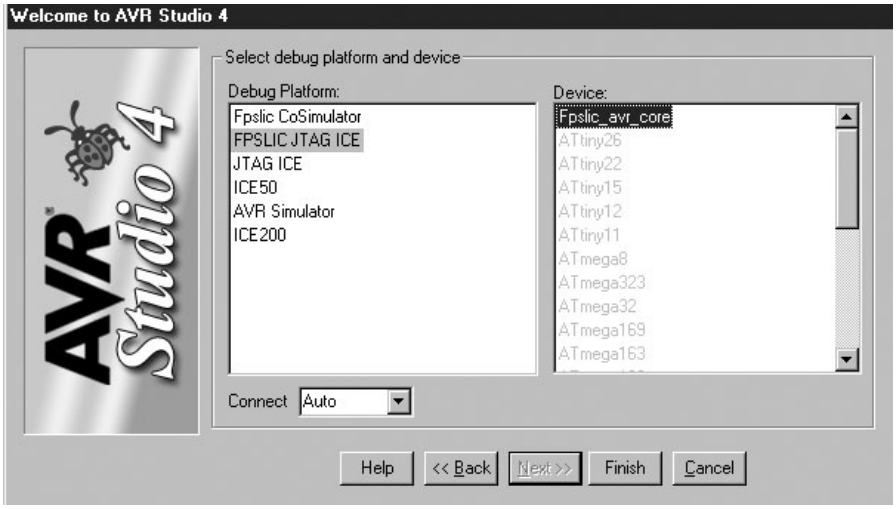
1. Go to *Programs > Atmel AVR Tools > AVR Studio 4*. A dialog box appears with two choices: *Create a New Project* or *Open* an existing project, see Figure 3-15.

Figure 3-15. New Project/Open Dialog Box



2. Make your choice and press *OK*, the *Select Debug Platform and Device* dialog box appears, see Figure 3-16.

Figure 3-16. Select Debug Platform and Device Dialog Box



3. Select *FPSLIC JTAG ICE* as the *Debugging Platform* and the *Fpslic_avr_core* device in order to proceed in FPSLICJTAG emulation.
4. Press *Finish*. AVR Studio will search all the COM ports for the JTAG ICE in a sequential manner. Make sure that other Atmel tools connected to the COM ports are switched off or disconnected.

   The JTAG ICE will then try to detect if there is power on the target board. If there is no power, a warning stating that the target board is switched off will appear. Turn on the power and retry.

   If there is power then the FPSLIC JTAG ICE will try to read the JTAG ID (JEDEC Identification Number) on the target device. If the JTAG ICE fails to read the JTAG ID, the user will be presented with one of the following dialogs:

   Warning: *The target FPSLIC is not connected to the JTAG ICE*.

   Workaround: Verify that the JTAG ICE is properly connected to the target board.

   Warning: *The JTAG enable (SCR27) and the On-Chip Debug enable (SCR26) bits in the System Control Register (SCR) are not set, thus disabling the OCD system*.

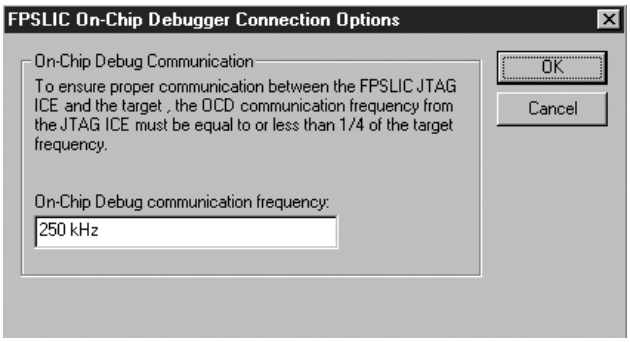
   Workaround: Use System Designer to reprogram the device and set the bits.

   Warning: *The code that is programmed in the device resets the debugging system or the Configuration Memory Lockout bit (SCR4) is set thus disabling the JTAG Interface.*

   Workaround: Use System Designer to reprogram the device and clear this bit.

   When the JTAG ID is read successfully, the *FPSLIC On-Chip Debugger Connection Options* dialog box appears, see Figure 3-17.

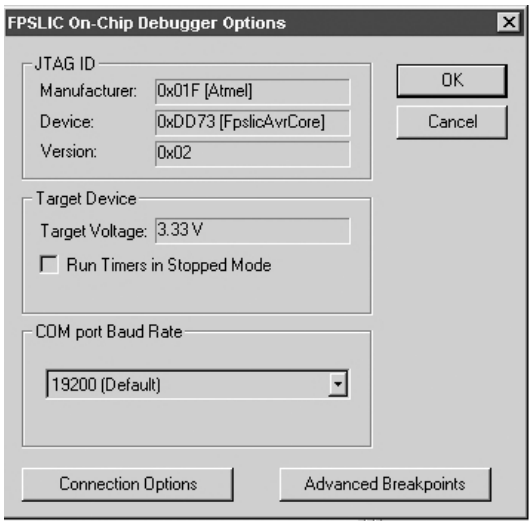
Figure 3-17. FPSLIC On-Chip Debugger Connection Options Dialog Box



To ensure proper communication between the JTAG ICE and the target FPSLIC, the OCD communication frequency from the JTAG ICE should be equal or less than 1/4 of the target FPSLIC frequency. It is important that the OCD communication frequency is set to be a maximum 1/4 of the frequency the target FPSLIC is running at. If the JTAG ICE communicates at a higher speed than the target FPSLIC can handle, communication will fail. If the OCD frequency is too low, communication will still work, but unnecessary delays are introduced.

5. Press *OK*. Based on the JTAG ID from the target FPSLIC, AVR Studio will now configure the correct I/O view and settings accordingly, see Figure 3-18.

Figure 3-18. FPSLIC On-Chip Debugger Options



To access the On-Chip Debugger Options while debugging, go to the *Debug* menu and select *FPSLIC JTAG ICE Options*.

The following is a short description of every option available to the FPSLIC JTAG ICE users.

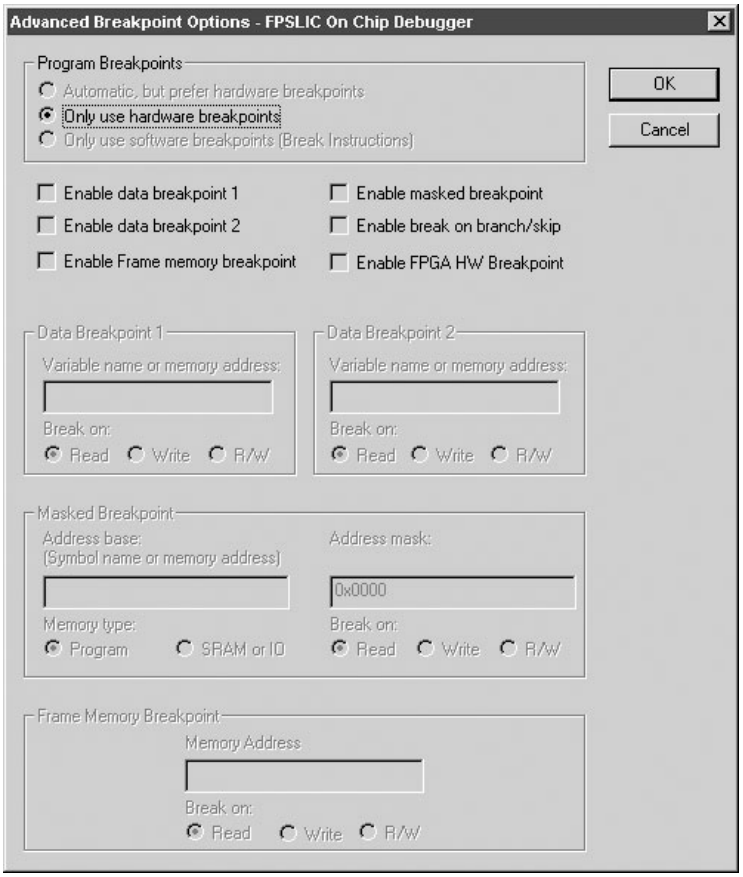
- JTAG ID. This box shows the device Manufacturer, the model and the version.
- Target Device. An informational option that displays the power Voltage of the target board as detected by the ICE.
- Run Timers in Stopped Mode. This mode allows the timers to continue running at their normal speed when single stepping through the code. When this box is not checked, the timers will increase the cycle according to the stepping through the code.

- COM and Baud Rate. Allows the user to select the preferred communication speed. Lower the baud rate if communication problems occur.

Note: If you set a Baud Rate in which the OCD cannot communication the JTAG ICE will not function correctly. Switch off and on the JTAG ICE, the target FPSLIC and AVR Studio and reopen them in order for the system to get the default Baud Rate value.

- Connection Options. Pushing this button will get the user to the *On-Chip Debugger Connections Options* dialog box shown in Figure 3-17.
- Advance Breakpoints. Pushing this button will get the user to the *Advanced Breakpoint Options* dialog box shown in Figure 3-19

Figure 3-19. Breakpoints Dialog Box



The following options are available:

- Program Breakpoints:
  - *Automatic, but prefer hardware Breakpoints*. In this mode AVR Studio will analyze the breakpoints and try to place the hardware breakpoints and the Break Instructions in an optimum manner.
  - *Only use hardware breakpoints*. Only the 3 general-purpose breakpoints are available. This is the default mode, and the only available mode for devices without break instruction support in JTAG ICE.

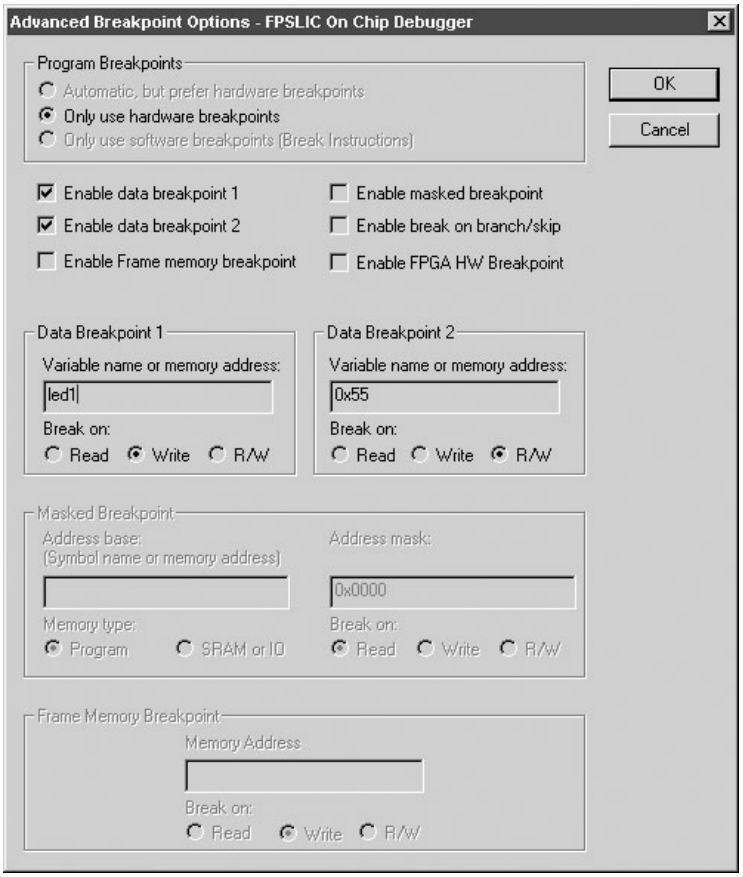
– *Only use Break instructions*. This mode forces AVR Studio to only use break instructions. No hardware breakpoints will be used. It is recommended to use one of the two other.

■ Enable Data Breakpoints

The data memory breakpoints, see Figure 3-20, can be set to one out of three modes:

– Data memory Read,
– Data memory Write, or
– Data memory Read or Write

Figure 3-20. Using Data Breakpoints



Data break points will work in I/O memory and SRAM only. It is not possible to set breakpoints in the Register file.

In order to use variable names (e.g. led1), the object file needs to include symbolic information. Using symbolic variables is possible if using a C compiler or an assembler that includes symbolic information. (e.g. EWBAVR, ICCAVR, IAR Assembler)

Note: The assembler provided with AVR Studio does not provide symbolic information. Some compilers do not provide symbolic information for I/O memory. Using absolute addresses will however always work.

When using Data breakpoints, the AVR will break after executing the instruction causing the break condition.

■ Enable Frame Memory Breakpoints

The FPSLIC OCD supports a single Frame (FPGA-SRAM) Memory break. Frame memory break moves the AVR core into the OCD stopped mode after a Frame memory read or write access (rising edge of frame clock) and before executing the new AVR instruction.

Three access modes are available to the Frame memory break:

– FPGA-SRAM Memory Read
– FPGA-SRAM Memory Write
– FPGA SRAM Memory Read/Write

Figure 3-21. Frame Memory Breakpoint

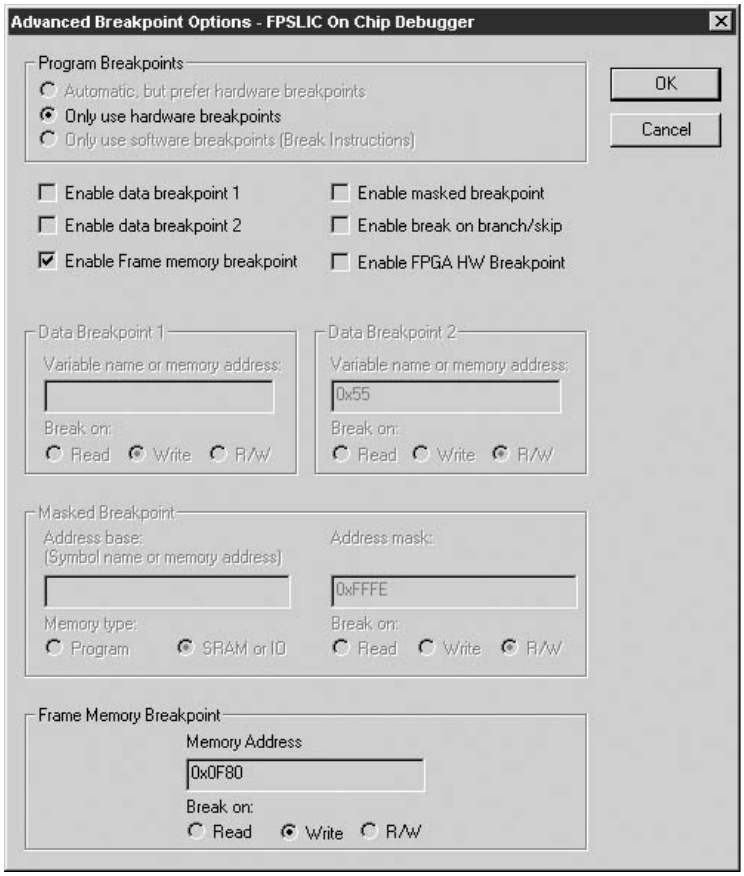
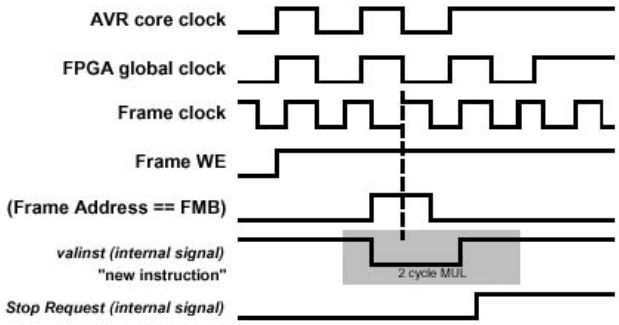


Figure 3-22 shows the relationship between a Frame Memory break (write access) asynchronous to the execution of a 2-cycle AVR instruction (e.g., MUL) and the AVR entering the OCD Stopped Mode.

Figure 3-22. Frame Memory Synchronization Delay (Write Access)

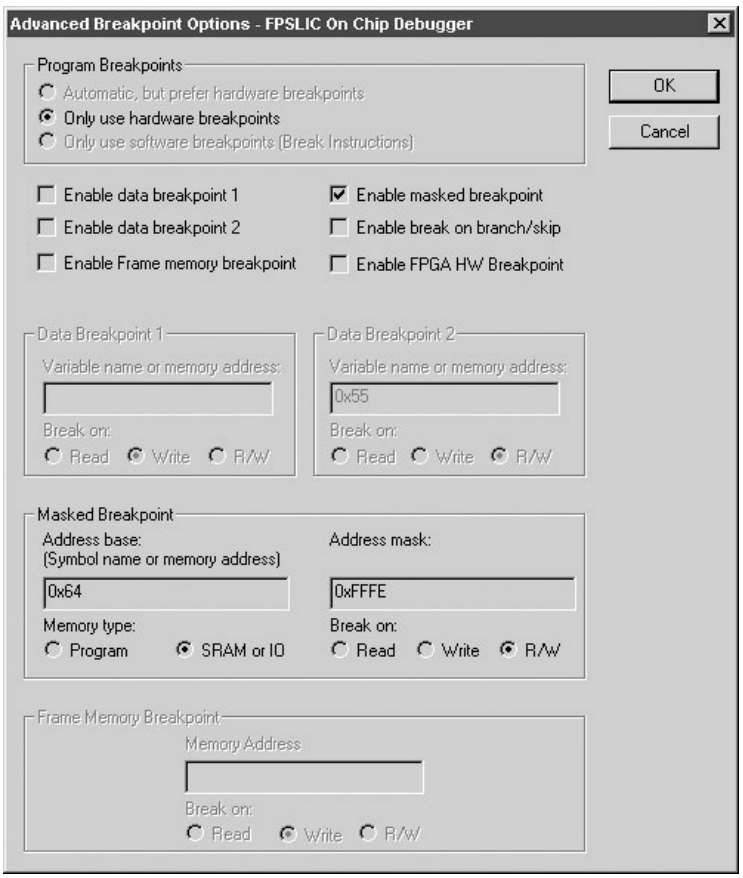


Note: Break on Frame Data content is not supported. The Frame Interface (SCR63) bit must be set to enable Frame access. The Cycle Counter, Time Elapsed, Frequency and StopWatch functionality is not available when using the FPSLIC JTAG ICE. This is a direct result of the nature of the implementation of the On-Chip Debug functionality.

- Enabling Masked Breakpoints:

When masked breakpoints are enabled you must specify the Address base and Address mask, see Figure 3-23. The two registers are bit-wise AND'ed together to generate valid break conditions. This value is compared to the masked Program Counter (PC) or Data Address to see if a valid break condition is present.

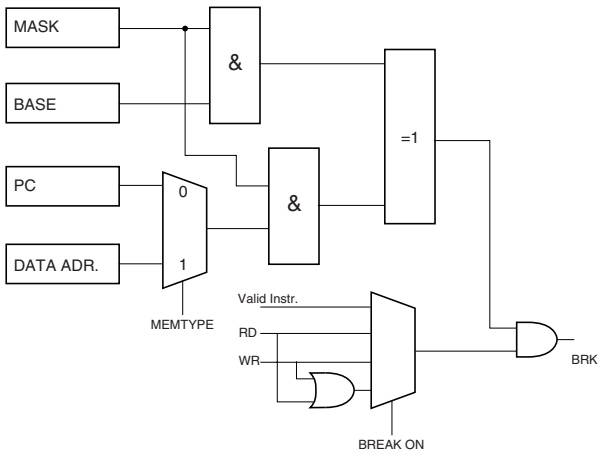
Figure 3-23. Using Masked Breakpoints



By setting a bit position in the mask to zero, that bit position will be “don’t-care” and will generate a valid break regardless if that bit position in the PC or data address is High or Low.

By setting a bit position in the mask to one, that bit position will be “locked” so that the corresponding bit position in the PC or data address must be at the same logic level as the bit position in the base address. Figure 3-24 shows the block diagram of how this is implemented.

Figure 3-24. Masked breakpoints block diagram



Consider the following examples:

Table 3-1. **Break Vector Examples**

Example	Address base	Address mask	Break Vectors	# Break Vectors
1	1010 1010 0101 0101	1111 1111 1111 1111	1010 1010 0101 0101	20 = 1
2	1010 1010 0101 0101	1111 1101 1111 1111	1010 10x0 0101 0101	21 = 2
3	1010 1010 0101 0101	1111 0000 1111 1111	1010 xxxx 0101 0101	24 = 16
4	1010 1010 0101 0101	1010 1010 0101 0101	1x1x 1x1x x0x0 x0x0	28 = 256
5	1010 1010 0101 0101	0000 0000 0000 0000	xxxx xxxx xxxx xxxx216 = 65536	

x = don’t care

Example 1

Setting all High in the Mask will result that only the base address will generate a valid break vector. Only valid Break vector is:

1010 1010 0101 0101

Example 2

Setting bit 9 in the Mask to zero will give two valid break vectors:

1010 1000 0101 0101

1010 1010 0101 0101

Example 3

Setting all bits in the mask to zero means that all addresses are valid break vector, and that it will single step the AVR core one instruction at the time.

The control logic will cause a break if the PC or Data address matches a valid break vector. The AVR core will cause a break after executing the instruction on the address causing the break condition.

- Enable Break on Branch/Skip

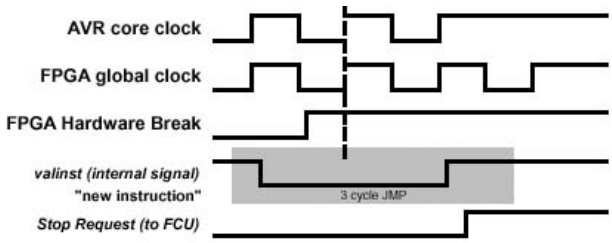
The *Enable break on branch/skip* can be enabled independently of how the 4 general-purpose breakpoints are used. A change of program memory flow is defined as any deviation from a linear flow, e.g.: when executing interrupts, Jumps, Branches, calls or skip instructions. A break on change of flow will break after executing the instruction causing the change of flow.

- Enable FPGA Hardware Breakpoint

The *FPGA hardware break* can be enabled independently of how the other Breakpoints are set. This breakpoint is an active High signal from the FPGA core into the FPSLIC OCD system. The signal is received asynchronously, but the generated stop request is synchronized by the OCD system to the AVR peripheral clock and gated by the availability of a new instruction. An FPGA hardware break moves the AVR core into the OCD Stopped mode before the execution of a new instruction

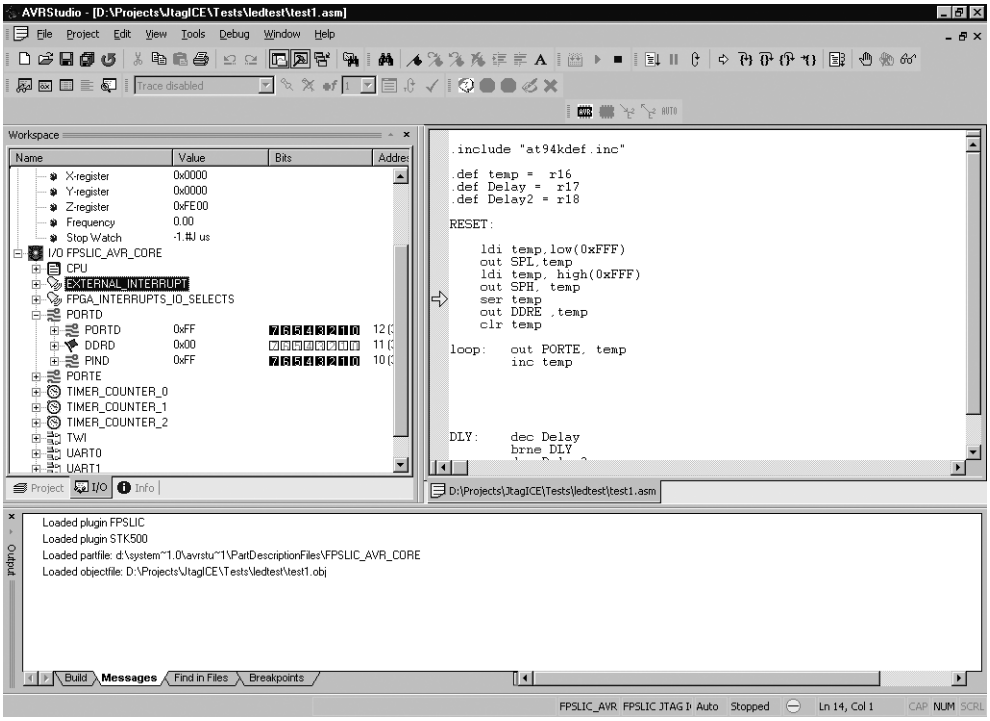
The next figure shows the relationship between the FPGA issuing a hardware break during the execution of a 3-cycle instruction (e.g. JMP) and the AVR entering the OCD stopped mode.

Figure 3-25. Synchronization Delay Following FPGA Hardware Break



6. Press *OK*. AVR Studio will now proceed to emulation, see Figure 3-26.

Figure 3-26. Emulation Example



3.4 Known Issues

When using the JTAG ICE some special considerations should be noted.

3.4.1 I/O Peripherals

All I/O peripherals will continue to run even though the program execution is stopped by a breakpoint.

Example: If a breakpoint is reached during a UART transmission, the transmission will be completed and the corresponding bits will be set. The TXC (transmit complete) flag will be set and will be available on the next single step of the code even though it normally would happen later in an actual device.

All I/O modules will continue to run in stopped mode with the following two exceptions:

- Timer/Counters
- Watchdog Timer

3.4.2 Timer/Counters in Stopped Mode

From the *On-Chip Debugging Options* menu, the user can decide if the Timer/Counters should continue to run in stopped mode or if they should be stopped.

3.4.3 Watchdog Timer

The watchdog timer will be halted in stopped mode. This avoids an unintentional AVR core reset during stopped mode.

3.4.4 Single Stepping

Since the I/O continues to run in stopped mode, care should be taken to avoid timing issues due to the fact that the I/O is running although the program execution is stopped.

Example:

```
OUT   PORTE, 0xAA
IN   TEMP,   PINE
```

In an actual device, this code would not read 0xAA because the data out has been latched to the physical pin and then latched back to the PIN register. A NOP instruction

must be placed between the OUT and the IN instruction to ensure that the correct value is present in the PIN register. In the JTAG ICE however, single stepping through this code will always give 0xAA in the PIN register since the I/O are not stopped but continue to run at full speed giving plenty of time to latch the data both out to the pin, and back again.

3.4.5 Software Breakpoints

Since a software breakpoint replaces the original instruction with a break instruction, the Flash page must be reprogrammed every time a software breakpoint is removed or added.

Extensive breakpoint settings will eventually wear out the flash memory. In addition, reprogramming the pages containing new breakpoint information will add to the startup time of the On-Chip Debugging.

For debugging sessions that require frequent changing of breakpoint positions avoid using software break instruction use hardware breakpoints instead. This will increase the speed, and the life of the device.

3.4.6 Target FPSLIC Reset and Power Down during Debugging

If the JTAG ICE loses its power during debugging, the communication will fail. It is only possible to continue debugging if the target FPSLIC loses its power.

However, pulling the target FPSLIC into Reset or Power Down while debugging may put the target device into another state from what AVR Studio and the JTAG ICE expects it to be.

Applying an operation in AVR Studio (Run, Single Step,) will update its view and response to reflect the state of the target AVR, but will discard whatever AVR Studio thought the prior position or state was. Note that if the target FPSLIC is powered down, it will temporarily forget its hardware breakpoints. The breakpoints will be updated in the target FPSLIC by applying a Run or Reset within AVR Studio. The hardware breakpoints will always remain in the target FPSLIC after a hardware Reset.

3.4.7 JTAG Relevant SCR settings

There are 2 bits in the System Configuration Register (SCR) that must be set for the JTAG ICE to function correctly. This is the JTAG enable (SCR27) and the OCD enable (SCR26). Also the Memory Configuration Lockout bit (SCR4) must be cleared or else the system will be in reset state.

3.4.8 Use AVR Studio 4.07 or Higher

There is no support for the FPSLIC JTAG ICE on any of the previous versions of AVR Studio.

3.4.9 The Message Window

The Message window gives important information on what AVR Studio and the FPSLIC JTAG ICE are doing.

3.4.10 JTAG Chain

It is not possible to run the JTAG ICE if the target FPSLIC device is connected in a JTAG chain. To ensure correct operation, the JTAG chain should not include other JTAG devices.

3.4.11 Alternative JTAG Pin Functions

If the JTAG interface is enabled, the JTAG pins cannot be used for alternative pin functions. They will remain dedicated JTAG pins.

Atmel Corporation

2325 Orchard Parkway
San Jose, CA 95131
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

Regional Headquarters

Europe
Atmel Sarl
Route des Arsenaux 41
Case Postale 80
CH-1705 Fribourg
Switzerland
Tel: (41) 26-426-5555
Fax: (41) 26-426-5500

Asia
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimshatsui
East Kowloon
Hong Kong
Tel: (852) 2721-9778
Fax: (852) 2722-1369

Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Atmel Operations

Memory
2325 Orchard Parkway
San Jose, CA 95131
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

Microcontrollers
2325 Orchard Parkway
San Jose, CA 95131
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

La Chantrerie
BP 70602
44306 Nantes Cedex 3, France
Tel: (33) 2-40-18-18-18
Fax: (33) 2-40-18-19-60

ASIC/ASSP/Smart Cards
Zone Industrielle
13106 Rousset Cedex, France
Tel: (33) 4-42-53-60-00
Fax: (33) 4-42-53-60-01

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Scottish Enterprise Technology Park
Maxwell Building
East Kilbride G75 0QR, Scotland
Tel: (44) 1355-803-000
Fax: (44) 1355-242-743

RF/Automotive
Theresienstrasse 2
Postfach 3535
74025 Heilbronn, Germany
Tel: (49) 71-31-67-0
Fax: (49) 71-31-67-2340

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Biometrics/Imaging/Hi-Rel MPU/
High Speed Converters/RF Datacom
Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
Tel: (33) 4-76-58-30-00
Fax: (33) 4-76-58-34-80

Atmel Programmable SLI Hotline
(408) 436-4119

Atmel Programmable SLI e-mail
fpslic@atmel.com

FAQ
Available on web site

e-mail
literature@atmel.com

Web Site
http://www.atmel.com

2451A–FPSLI–03/04 /xM