
HDLPlanner[®]: Design Development Environment for HDL-based FPGA Designs

Abstract

Rapid prototyping of designs using FPGAs requires HDL-based design entry which leverages upon highly parameterized components in conjunction with an ability to translate these components to layouts. Designs that are developed with layout related considerations more often tend to become technology specific. This has a direct consequence in migrating designs to ASICs and may require a design to go through an additional design cycle. HDLPlanner solves this problem by providing an environment for creating technology independent HDL designs, which when translated to Atmel FPGAs, are guaranteed to produce performance delivering layouts. The primary contribution of the HDLPlanner is its ability to address layout considerations up front in the design process without compromising the technology independence of the design. In doing so, the planner partitions the design creation process between itself and the synthesis software in the following way. It manages layout related considerations by itself and delegates the task of random logic optimization to synthesis software. The planner is tightly integrated with MGL Parameterized Layout Synthesis Software [1] which produces highly efficient layouts of the components.

Motivation

Hardware description languages (HDL) are increasingly replacing schematic oriented design entry methods because they offer a very short and efficient

design cycle involving simulation, synthesis and testing.

From a synthesis perspective, the most appealing benefits of using HDLs are the ability to parameterize modules and the technology independent manner in which designs can be created. Module parameterization encourages reusing existing design effort and offers a convenient and powerful method for customizing modules in the environment in which they are used. Another important advantage of using an HDL centric approach is to leverage upon the design optimization support that is built into all HDL tools. This optimization support allows rapid searching for a design solution that meets user's constraints in an automated, correct by construction fashion. All these benefits make HDLs a preferred choice for designers involved in rapidly prototyping digital systems using FPGAs.

HDL centric approach, on the other hand, can result in severe penalty if the synthesis and optimization support that is built into it is used without giving a due consideration to underlying technology. This is particularly true in the case of FPGAs because synthesis tools cannot incorporate FPGA technology specific issues such as logic cell, RAMs, tri-state resources, clock and reset scheme etc. during synthesis, to produce good quality designs (these issues are dealt in the later part of this section). The outcome can be a large netlist which may not be placed and routed in an optimum fashion.

Some recent improvements in synthesis technology have partially overcome these shortcomings. A notable improve-



Design Development for HDL-based FPGA Designs

Application Note

Rev. 1444A-10/99



ment called operator inferencing, identifies operator functions from HDL expressions and links them to their preferred layout implementation instead of constructing their

gate level representations. In many cases, however, operator inferencing does not guarantee optimal results, a case for which is given in Figure 1.

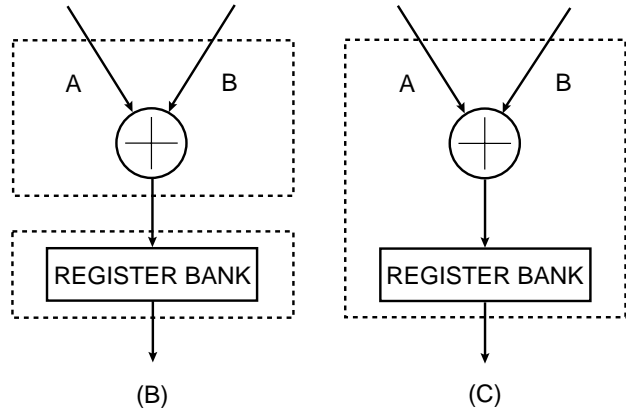
Figure 1. (a) VHDL template of registered adder. (b) Optimized implementation and (c) preferred implementation.

```

PROCESS (CLOCK, RESET)
BEGIN
    IF (RESET = 'B0') THEN
        SUM <= "00000000";
    ELSE IF (CLOCK = '1' AND CLOCK EVENT)
        SUM <= A + B;
    END IF;
END PROCESS;

```

(A)



The operator inferencing has one more limitation- only operators that are supported in the language can be inferred. Simple building blocks such as counters and decoders cannot be inferenced and must be explicitly used in the design. Explicit references to cells can cause design portability problem.

In addition to the shortcomings in synthesis technology as discussed above, issues that are specific to FPGA technology and limitations that are inherent to HDL languages and synthesis systems are discussed below.

FPGA Technology Specific Considerations

Synthesis tools perform an architecture specific optimization, without giving due consideration to the technology contents of the FPGA device ⁽¹⁾.

Note: 1. The term FPGA architecture is broadly used to identify architecture of the basic logic cell in it.

What is really desired from the performance standpoint is to accommodate technology specific resources in the design synthesis. Examples of such resources are clock and reset resources, tri-state resources, wired logic resources, I/O buffers, on-chip configurable memory resources and their address decoding circuitry etc. A cost driven optimization of these resources is important to uncover optimum performance from an underlying technology. We are not aware of any synthesis system which can fully understand and utilize technology specific information to perform constraint driven synthesis.

The limiting factors described above leave designers very limited options for maintaining technology independence

and a search for an optimal solution often requires manual intervention in the design process.

Limitations in Languages and Software Tools

Some limitations in HDLs, as described below can have a significant bearing on the quality of the synthesis results. An example of these limitations is the inability to parameterize design modules with respect to clock, set and reset schemes (i.e., according to active edges and polarities, respectively). In the absence of this capability, re-using modules can be a difficulty to achieve. In some cases, lack of the provision is also likely to create designs which may indiscriminately use limited numbers of clock and reset resources available on FPGAs.

Besides the problem outlined above, CAE vendors do not seem to have a consistent method for supporting parameterization in Verilog. For example Synopsys recommends using the parameter construct and Exemplar recommends using the defparam construct to define the parameters. This difference can cause design migration problems between synthesis platforms.

Proposed Solution

Atmel Corporation has developed a design planning environment called HDLPlanner that encourages users to follow meet-in-the-middle methodology for creating HDL designs. HDLPlanner can be considered as a design manager tool which enforces design partitioning and generates instructions to synthesis and placement software towards an objective of creating highly regular and efficient implementations. It does so by encouraging designers to partition the system into a set of basic building blocks. It

then provides highly parameterized behavioral HDL descriptions for them and a seamless link to generating layout descriptions via MGL Parameterized Layout Synthesis software. MGL generates layouts that are highly optimized to architecture and technology. In this fashion, layout-specific issues can be addressed much earlier in the design process in a technology independent fashion, resulting in a simplified and shortened design cycle.

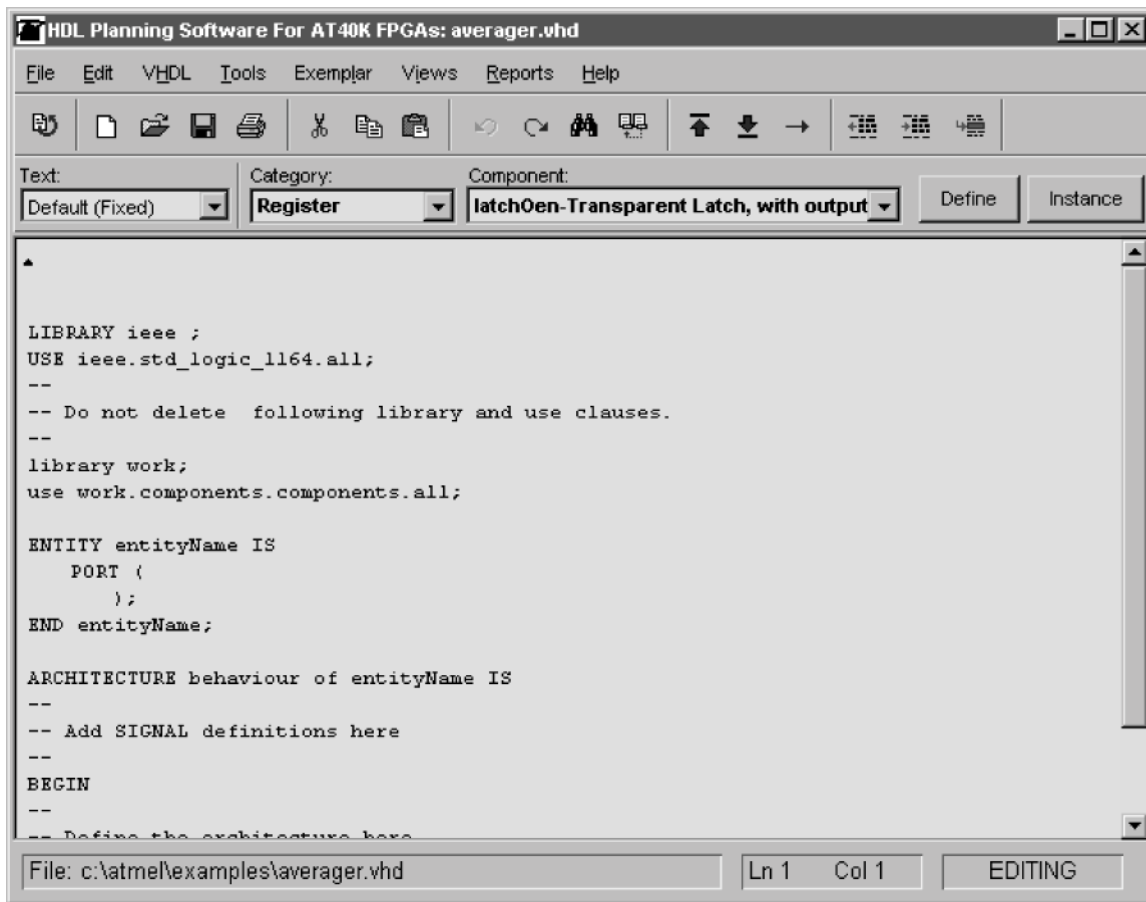
HDLPlanner has special code generation support with which modules can be parameterized with respect to clock and reset pins. It is also tightly integrated with the target synthesis tool and knows the inner workings of each tool such as the parameterization and elaboration mechanisms.

The remainder of this paper is organized as follows. Section II describes the Graphical User Interfaces of HDLPlanner. Section III describes the central ideas in planning HDL designs using the planner. Section IV presents the features and benefits of using HDLPlanner. Section V discuss future work and presents the conclusion.

HDLPlanner: An Environment for Design Planning

The HDLPlanner Graphical User Interface is described briefly.

Figure 2. HDLPlanner Graphical User Interface

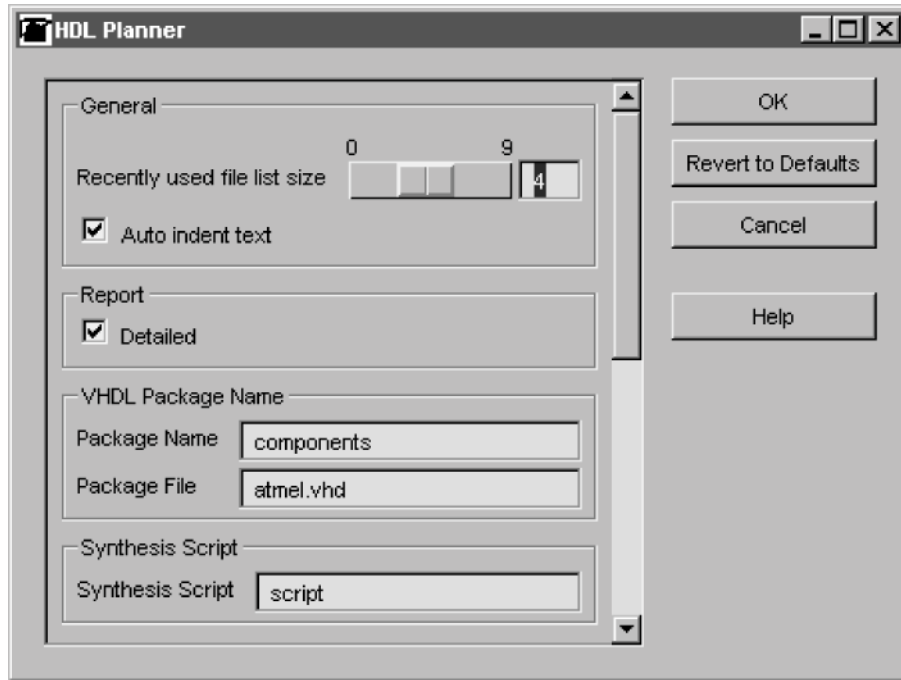


DESIGN EDITOR: The built-in editor contains buttons for project creation, saving and supports basic text editing operations such as cut, copy, paste, search, find etc. These buttons are located on the tool bar.

MODULE DEFINITION AND INSTANTIATION PANES: Special list panes and buttons are provided to select, define and instantiate modules (Refer to Figure 2). A fully

parameterized behavioral description of the module can be inserted into text view by pressing the **Define** button. The **Instantiate** button can be used to instantiate a component. The instantiation statement conforms to the parameterization mechanism recommended by the synthesis software being used.

Figure 3. Options window for parameterizing designs for clock and reset polarities.

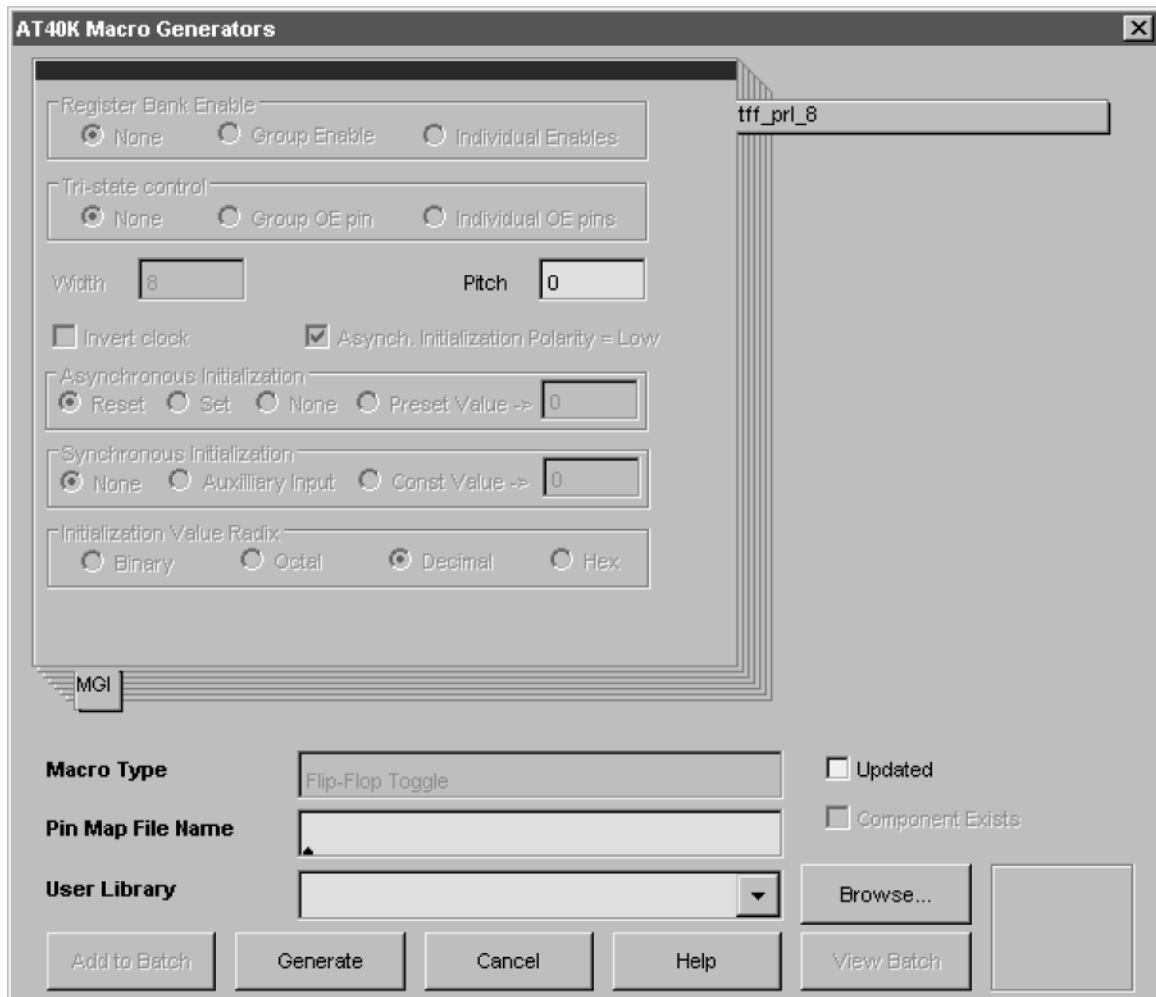


OPTIONS: The Options user interface (UI) in Figure 3 can be invoked by selecting the **Tools > Options** sub menu. The important options are listed below.

- **Clock trigger:** Specify clock trigger for the component being defined/instantiated.
- **Clock signal:** Specify global signals to be used in buffer insertion.
- **Set or reset polarity:** Specify polarity of the set or reset pin for the component being defined/instantiated.
- **Set or reset:** Specify if component has asynchronous set or reset.
- **Synthesis script:** Specify the name of the synthesis script generated automatically by HDLPlanner.
- **Top level design:** Specify the name of the top level design. It is used in the synthesis script to set the top level design name.
- **VHDL package name:** Specify the name of the package in which all defined components will be declared. This package is saved in the specified file name. This option is specific to VHDL only.

LAYOUT SYNTHESIS: A menu button **Tools > Invoke Macro Generators** is provided to generate FPGA layouts for all components that are used in the design. Refer to Figure 4 for the Macro Generator UI. In this UI, users have the opportunity to specify floorplan-specific parameters such as **pitch** and folding options with which a layout is to be generated.

Figure 4. A sample of a macro generator user interface.



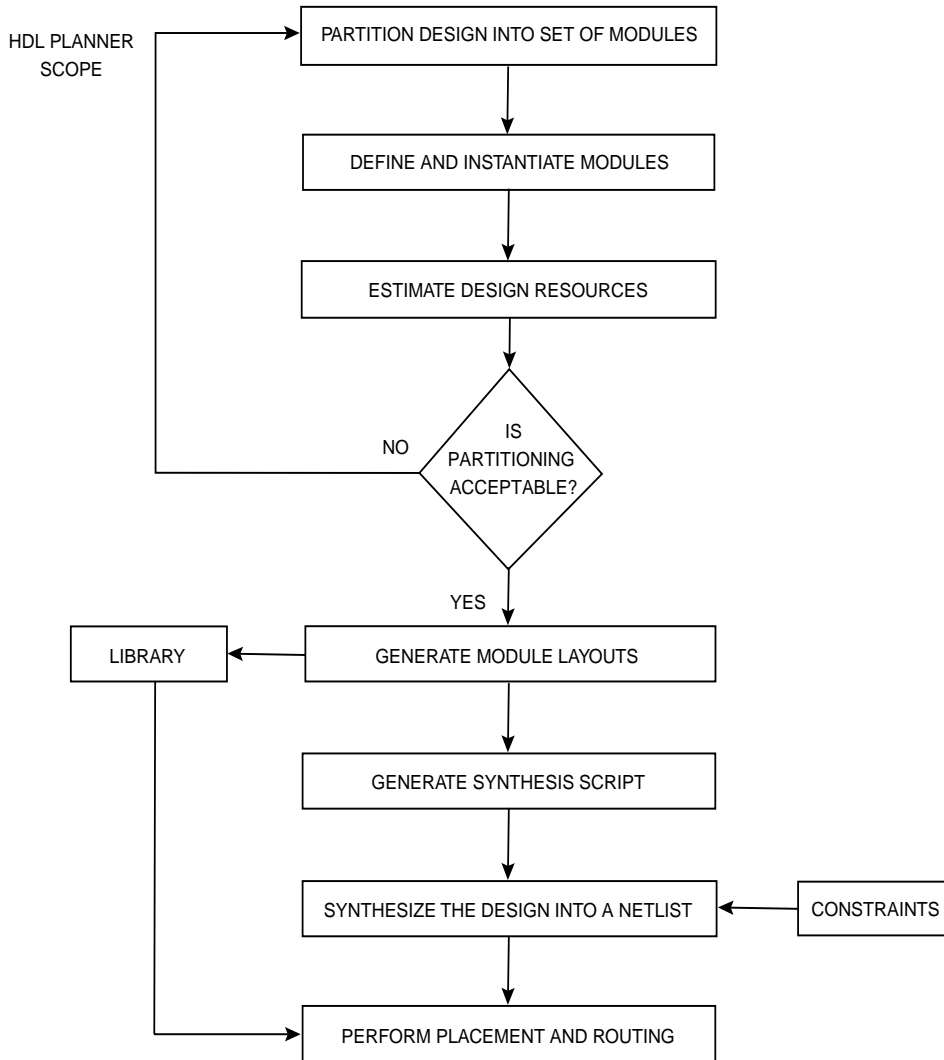
HDLPlanner: Planning Designs at HDL Level

In FPGA technology, design planning contains the following key elements.

1. **SYSTEM PARTITIONING:** Identifying basic components contained in the system
2. **CLOCK AND RESET SCHEME:** Deciding upon a preferred clock and reset scheme for the design
3. **COMPONENT GENERATION AT HDL LEVEL:** Defining components (i.e. generating their behavioral descriptions) consistent with the set / reset scheme
4. **COMPOSING THE DESIGN:** Construct the top level design by instantiating components in step 3
5. **LAYOUT SYNTHESIS:** Generate hard layouts for the components by supplying layout specific parameters such as pitch, layout folding, and layout optimization options which create speed critical or area critical layouts.

This process is illustrated in Figure 5.

Figure 5. Design Planning using HDLPlanner



Summary

The benefits and features of the HDLPlanner software are summarized below.

Design Entry Specific

DESIGN EDITING: HDLPlanner has an editing environment for planning, entering and maintaining HDL descriptions. It has a comprehensive set of pre-verified templates of complex HDL constructs that can be used to speed up design entry.

TECHNOLOGY INDEPENDENT DESIGN ENTRY: Designs created in HDLPlanner are completely technology independent, conform to vendor laid out synthesis guide-

lines, and contain a complete simulation model within them (contains no black boxes).

DESIGN REUSE: HDLPlanner has a User Interface to easily define and instantiate pre-verified functional modules. These modules can be parameterized for bit-widths and also clock and reset schemes.

Technology Specific

LINKS TO LAYOUTS: The layout synthesis software interface accessed from HDLPlanner translates functional modules to layouts that are highly optimized for the FPGA architecture and technology.

MANAGEMENT OF CLOCKING AND RESET

RESOURCES: HDLPlanner simplifies the task of managing of clock and set/reset resources on the FPGA. Parameterization around clock and reset resources is not supported by any of the HDL languages.

Synthesis Tools Specific

OVERCOMES LIMITATIONS IN SYNTHESIS TECHNOLOGY: HDLPlanner partitions the task of design optimization between itself and the synthesis software, allowing each to focus on the job it can do the best.

TIGHTLY INTEGRATED WITH SYNTHESIS TOOLS: The synthesis script generated by HDLPlanner contains tool-specific directives that are useful for efficient synthesis.

Productivity Specific

PERFORMANCE ESTIMATION: On line reports and statistics of modules aid in arriving at reasonable performance estimates quickly.

SHORTER DESIGN CYCLE: Pre-verified, re-usable components and automatic template generation cuts down the length of the design cycle to a minimum.

Software Architecture Specific

COMPLETELY TRANSPARENT: HDLPlanner is an open system. Users can integrate their components and use them in their design process as if they were shipped from Atmel.

KNOWLEDGE ARCHIVAL: HDLPanner has an open help system, allowing users to document their synthesis experiences and refer to them or share them with others at a later date.

Future Work and Conclusion

Many enhancements are being developed for the HDL-Planner. There is a need to support behavioral timing analysis for precise timing estimates. Support for generating state machine descriptions and generating VHDL/Verilog test benches are also planned in future releases.

Here we have presented an HDL planning software, HDL-Planner. This software can be used for creating technology independent designs. The software has many key features that help in reducing design development cycle time and produce efficient designs for the given target technology. Presently only Atmel's AT6000 and AT40K FPGA families are supported.



Atmel Headquarters

Corporate Headquarters

2325 Orchard Parkway
San Jose, CA 95131
TEL (408) 441-0311
FAX (408) 487-2600

Europe

Atmel U.K., Ltd.
Coliseum Business Centre
Riverside Way
Camberley, Surrey GU15 3YL
England
TEL (44) 1276-686-677
FAX (44) 1276-686-697

Asia

Atmel Asia, Ltd.
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimhatsui
East Kowloon
Hong Kong
TEL (852) 2721-9778
FAX (852) 2722-1369

Japan

Atmel Japan K.K.
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

Atmel Colorado Springs

1150 E. Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
TEL (719) 576-3300
FAX (719) 540-1759

Atmel Rousset

Zone Industrielle
13106 Rousset Cedex
France
TEL (33) 4-4253-6000
FAX (33) 4-4253-6001

Fax-on-Demand

North America:
1-(800) 292-8635
International:
1-(408) 441-0732

e-mail

literature@atmel.com

Web Site

<http://www.atmel.com>

BBS

1-(408) 436-4309

© Atmel Corporation 1999.

Atmel Corporation makes no warranty for the use of its products, other than those expressly contained in the Company's standard warranty which is detailed in Atmel's Terms and Conditions located on the Company's web site. The Company assumes no responsibility for any errors which may appear in this document, reserves the right to change devices or specifications detailed herein at any time without notice, and does not make any commitment to update the information contained herein. No licenses to patents or other intellectual property of Atmel are granted by the Company in connection with the sale of Atmel products, expressly or by implication. Atmel's products are not authorized for use as critical components in life support devices or systems.

Marks bearing ® and/or ™ are registered trademarks and trademarks of Atmel Corporation.

Terms and product names in this document may be trademarks of others.



Printed on recycled paper.

1444A-10/99/xM