Programmable System Level Integration brings System-on-Chip Design to the Desktop

Joel Rosenberg
PSLi Product Line Director
Atmel Corporation
2325 Orchard Parkway
San Jose, California 95131 USA

In the mid-1980s the gate array offered logic designers the promise of thousands of gates of logic all on a single chip. The appeal of the gate array was to reduce the amount of board area, lower power consumption, improve reliability and reduce overall system cost. Unfortunately the benefits afforded by gate arrays were offset by the high cost of design tools, the non-recurring engineering charges (NRE) every time a design revision was made, the high degree of design risk, and the extended lead-time from design to production silicon. As such gate arrays, with all their promise, were a feasible solution to a few large companies who could afford the up front costs of tools and NREs and manufacture products in very large volumes.

By 1985 the first Field Programmable Gate Arrays (FPGAs) were introduced. Xilinx developed a family of standard user-configurable products utilizing industry standard SRAM technology, and a (relatively) low cost design tool that for the first time made Gate Array technology available to virtually any design engineer who had access top a desktop computer.

Today the chip industry has progressed from putting thousands of gates of logic into a single chip to putting an entire system on a chip (SOC). This means integrating not only the programmable and ASIC logic on a single IC, but also including the processor, memory and analog functions as well, as shown in figure 1.

Like their gate array predecessors in the 1980s and 1990s, System-On-Chip capability is extremely expensive and out of reach for most design projects today. Design tools, including co-verification, can cost in excess of $100K per seat; intellectual property cores are expensive and largely unproven, the mask charges for a 0.25u array are in excess of $250K, and the system level knowledge and experience required to successfully implement a SOC is not readily available. These issues are summarized in figure 2.

A Typical System Board

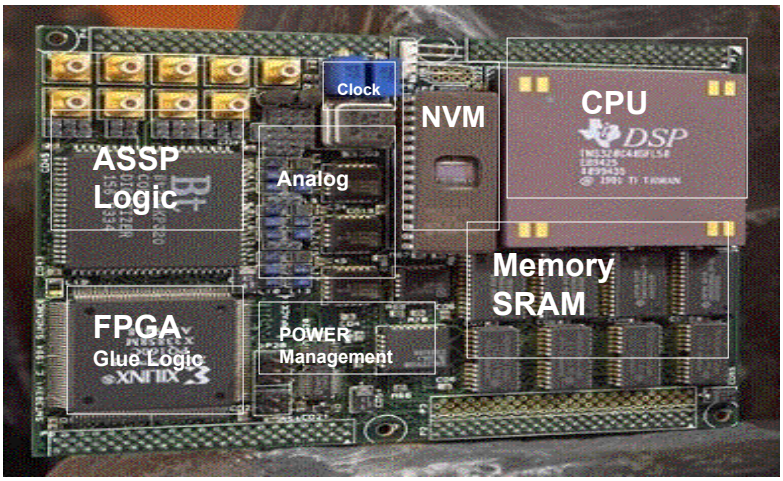


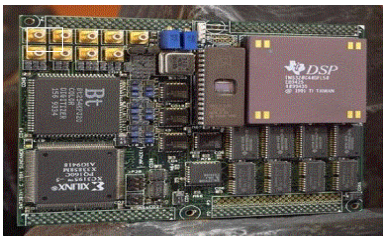
Figure 1

FPSLIC Architecture

A new family of standard products, called FPSLIC (Field Programmable System Level Integration Circuits) and associated design tools have been introduced to address programmable SOC. A single FPSLIC (shown in figure 3) contains embedded 5K - 40K gates FPGA, microcontroller, Memory, peripherals and interface logic. The low cost design tools include a system design manager, synthesis, hardware and software simulation, and Co-verification. These products, along with the tools, bring the promise of SOC to all design engineers, much in the same way that FPGAs brought the power of the gate array to the desktop of thousands of designers over the last several years.

System-On-Chip Issues

- **$250K+ NRE**
- **$100K+ design tools**
- **Large volume requirements**
- **Custom product**
- **Long design time**
- **Extensive design experience**
- **High risk**
- **IP issues (availability, cost implementation)**



>> System level integration not viable for most customers

Figure 2

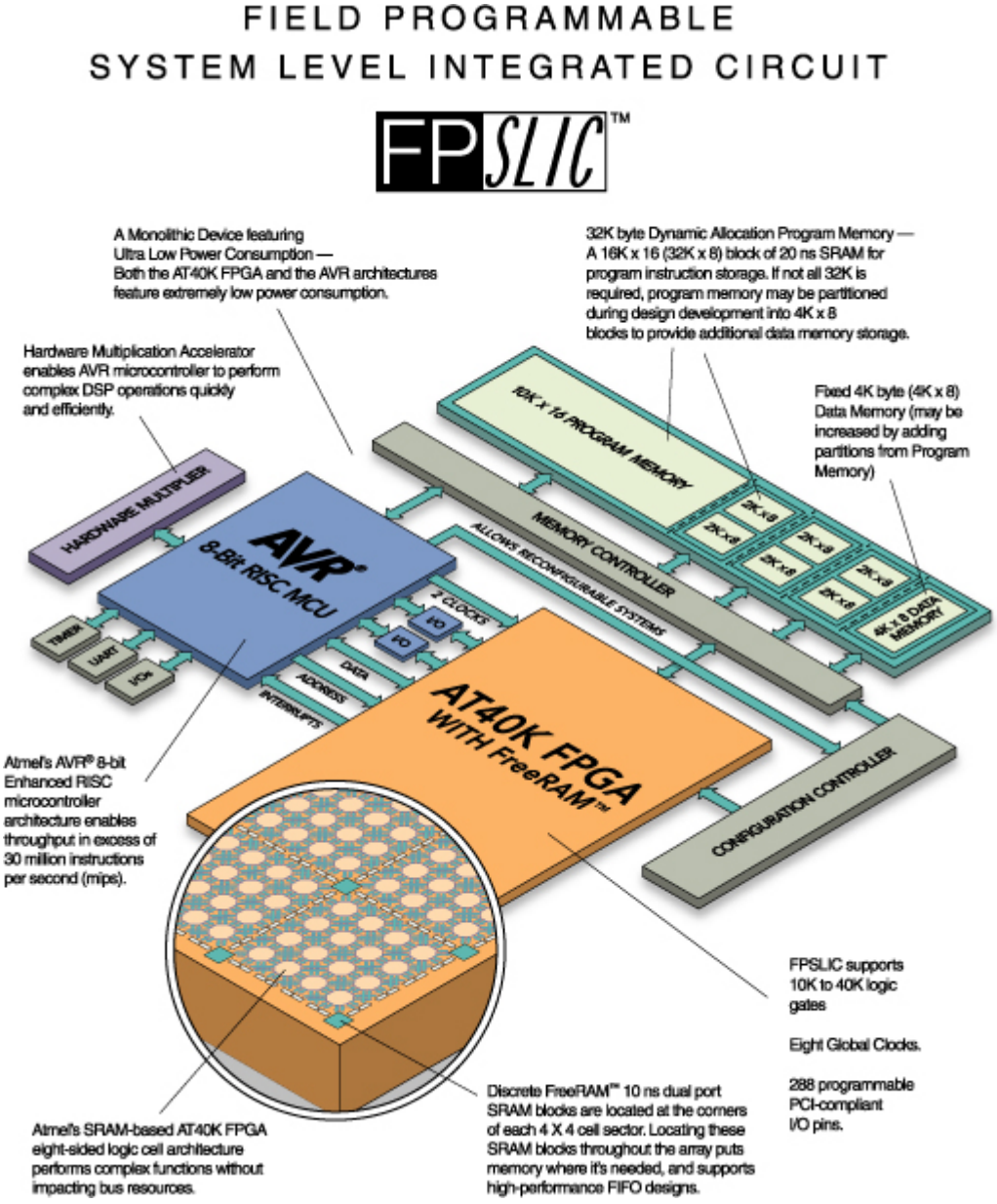

FIELD PROGRAMMABLE
SYSTEM LEVEL INTEGRATED CIRCUIT
FPSLIC™
A Monolithic Device featuring
Ultra Low Power Consumption —
Both the AT40K FPGA and the AVR architectures
feature extremely low power consumption.
32K byte Dynamic Allocation Program Memory —
A 16K x 16 (32K x 8) block of 20 ns SRAM for
program instruction storage. If not all 32K is
required, program memory may be partitioned
during design development into 4K x 8
blocks to provide additional data memory storage.
Hardware Multiplication Accelerator
enables AVR microcontroller to perform
complex DSP operations quickly
and efficiently.
Fixed 4K byte (4K x 8)
Data Memory (may be
increased by adding
partitions from Program
Memory)
HARDWARE MULTIPLIER
AVR®
8-Bit RISC MCU
MEMORY CONTROLLER
ALLOWS RECONFIGURABLE SYSTEMS
4K x 8 DATA MEMORY
TIMER
UART
I/Os
2 CLOCKS
I/O
DATA
ADDRESS
INTERRUPTS
AT40K FPGA
WITH FreeRAM™
CONFIGURATION CONTROLLER
Atmel's AVR® 8-bit
Enhanced RISC
microcontroller
architecture enables
throughput in excess of
30 million instructions
per second (mips).
FPSLIC supports
10K to 40K logic
gates
Eight Global Clocks.
288 programmable
PCI-compliant
I/O pins.
Atmel's SRAM-based AT40K FPGA
eight-sided logic cell architecture
performs complex functions without
impacting bus resources.
Discrete FreeRAM™ 10 ns dual port
SRAM blocks are located at the corners
of each 4 X 4 cell sector. Locating these
SRAM blocks throughout the array puts
memory where it's needed, and supports
high-performance FIFO designs.

Figure 3

The embedded microcontroller core is the Atmel AVR. It is an 8-bit RISC processor that is capable of executing most of its 120+ instructions in a single clock cycle. The AVR is very code efficient when compared to other 8-bit processors, and when embedded in an SRAM-based FPSLIC device ultimately creates 3 significant advantages – much higher throughput, much lower power consumption and less memory required to store the uC instruction set.

Embedded within the FPSLIC devices are several fixed peripherals associated with the AVR; including a high speed 10-bit multiplier, 2 UARTs, watch-dog timer, programmable timer/counters and interrupts. Additional custom peripheral functions can be programmed into the FPGA and mapped directly into the AVR address space. The AVR can actually reconfigure the entire contents of the FPGA, as well as individual elements of the FPGA as shown in figure 4. This FPGA capability, called Cache Logic™, is analogous to the idea of cache memory, but now those logic functions that are currently active, or require very high performance, are implemented in the FPGA, while those functions that are not currently being used are stored in less expensive non-volatile memory, as shown in figure 5. These FPSLIC features enable reconfigurable peripherals on-demand, as well as adaptive hardware including logic acceleration, adaptive filters for DSP, reconfigurable cross point switches and reconfigurable network processors, and many other possibilities.

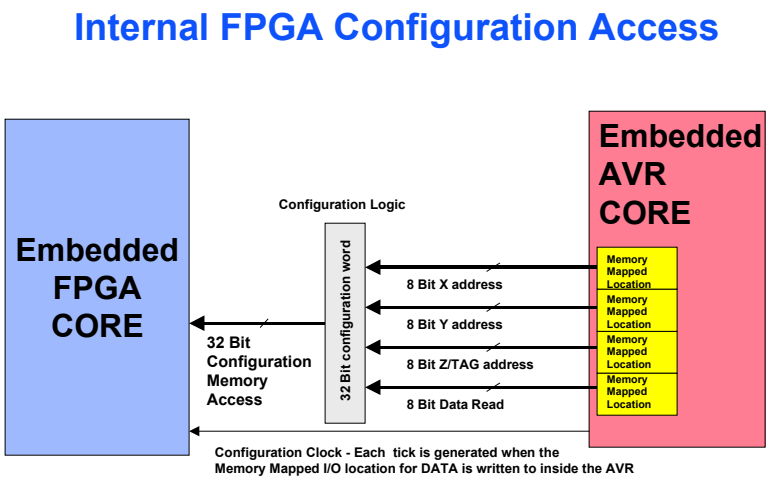


Figure 4

In addition, the interface between the FPGA and the AVR has been implemented in ASIC gates to eliminate the need to use valuable FPGA gates for this function. The same is true of the memory interface and controller. Figure 6 provides details on the AVR and FPGA interface, showing how peripheral functions implemented in the FPGA can be directly mapped into the address space of the AVR microcontroller. Again, because the AVR microcontroller and FPGA interface logic is "hardwired" into the FPSLIC device, the effective utilization of the FPGA logic gates is significantly higher. The result of these features is to increase the efficiency of the FPGA, improve performance, and simplify the design process.

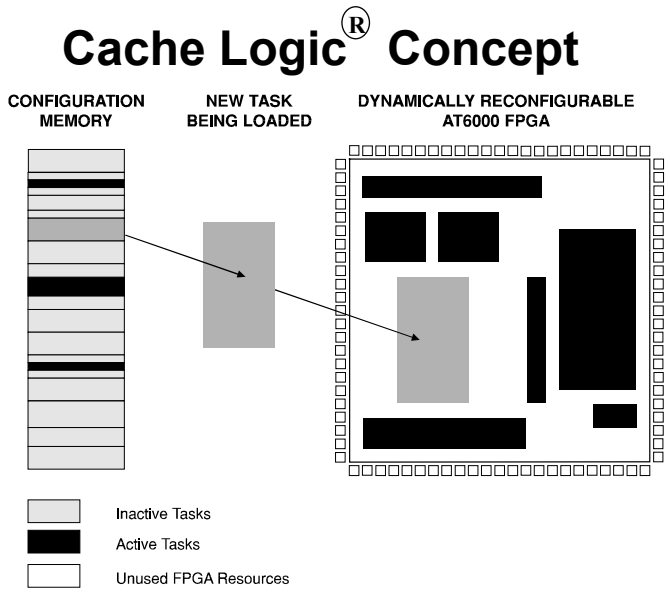


Figure 5

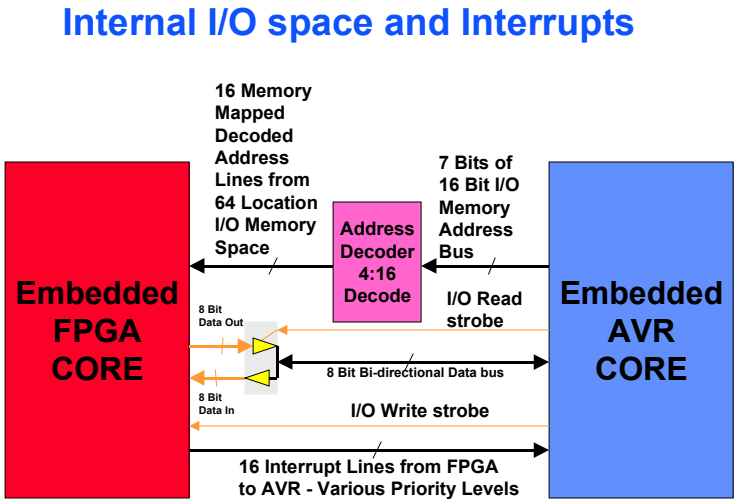


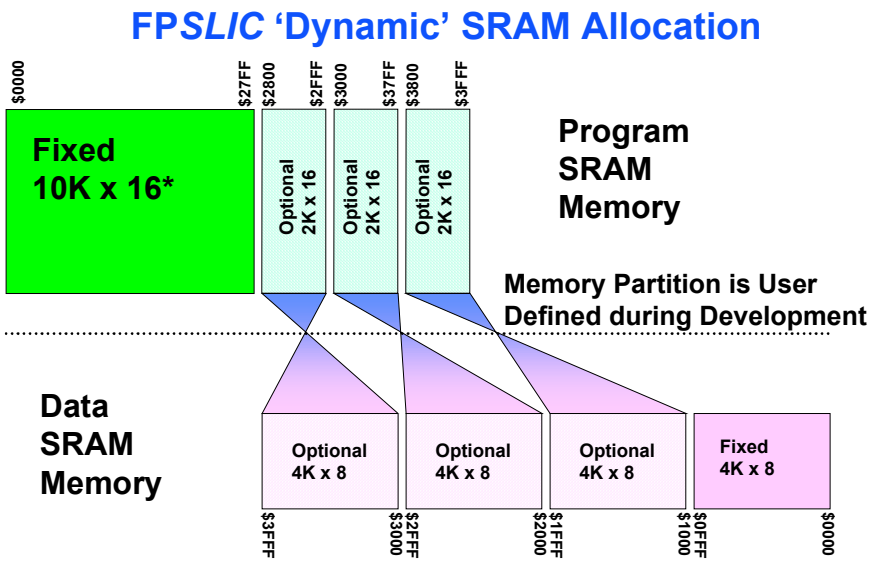
Figure 6

The embedded FPGA contains all of the features of Atmel's AT40K FPGA. The AT40K is a family of fully PCI-compliant, SRAM-based FPGAs with distributed 10ns programmable synchronous/asynchronous, dual port/single port SRAM, 8 global clocks, Cache Logic ability (partially or fully reconfigurable without loss of data), automatic component generators, and range in size from 5,000 to 150,000 usable gates. I/O counts range from 128 to 384 in industry standard packages ranging from 84-pin PLCC to 475-pin BGA, and support 3V and 5V designs.

The AT40K is designed to quickly implement high performance, large gate count designs through the use of synthesis and schematic-based tools used on a PC, Sun and HP platform. Atmel's design tools provide seamless integration with industry standard tools from Cadence (Concept/Verilog), Everest, Exemplar, Mentor, OrCAD, Synario, Veribest, and Viewlogic.

The AT40K can be used as a Coprocessor for high-speed (DSP/Processor-based) designs by implementing a variety of compute-intensive, arithmetic functions. These include adaptive finite impulse response (FIR) filters, fast Fourier transforms (FFT), convolvers, interpolators and discrete-cosine transforms (DCT) that are required for video compression and decompression, encryption, convolution and other multimedia, telecom and industrial control applications.

In addition to the AT40K's distributed FreeRAM SRAM, the FPSLIC device contains a large block of SRAM that can be configured at program memory for the AVR microcontroller and data memory for the FPGA and AVR, as shown in figure 7.



* The AT94K05 includes a 2K block.

Figure 7

All FPSLIC™ devices include a fixed block of 10Kx16-program memory and a fixed block of 4Kx8 data memory. There are additional blocks that can be programmed (at run time) to increase program memory to 16Kx16 – 8K x 18 on the AT94K05 – (in incremental blocks of 2Kx16) and/or increase data memory to 16Kx8 (in incremental blocks of 4Kx8). The programmable instruction and data memory size enables the memory allocation to be optimized for multiple hardware and software configurations. The data memory is analogous to dual port RAM, in that both the FPGA and the AVR, as shown in figure 8 can access it.

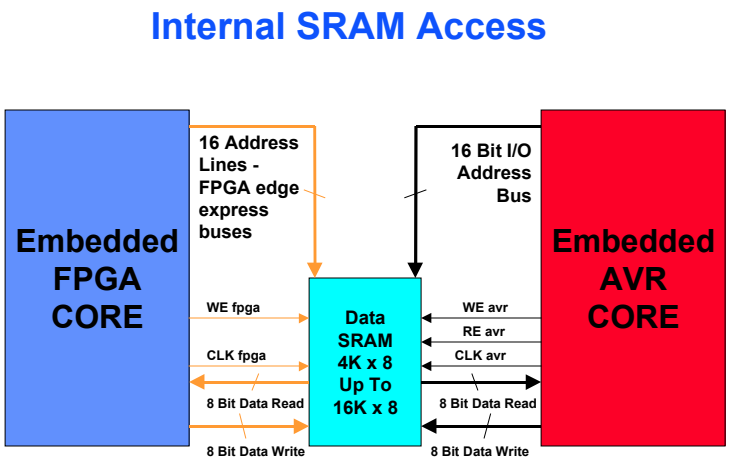


Figure 8

FPSLIC represents a family of products, not just a single device. The initial FPSLIC family is shown in figure 9. There are 3 initial devices, each containing identical features; with the exception of the number of available FPGA gates, associated FreeRAM and Program/Data memory. The usable FPGA gate densities range from 5,000 to 40,000 usable gates. In fact the effective FPGA gates available are substantially higher due to the interface between the FPGA, memory and the microcontroller already being implemented in ASIC gates, leaving more space in the FPGA for implementation of peripherals and other value added logic.

Device	Samples	FPGA Gates	FreeRAM™ SRAM	FPGA I/O	Program/Data SRAM
AT94K10	Now	10K	4608 bits	upto 204	20K –32K Bytes/ 4K-16K Bytes
AT94K20	Now	20K	8192 bits	upto 298	20K –32K Bytes/ 4K-16K Bytes
AT94K40	Now	40K	18432 bits	upto 384	20K –32K Bytes/ 4K-16K Bytes

Figure 9

The benefits of the FPSLIC architecture and integration over currently available discrete solutions include:

- >50% reduction in PCB area
- >50% reduction in power consumption
- >200% performance improvement
- Reduced design complexity
- Improved reliability
- 1-3 months improvement on time-to-market

FPSLIC Design Tools

While the concept of system-on-chip design implies many benefits, one of the major drawbacks today is the cost of the tools and engineering charges required to build such products. The price of co-verification tools today alone is typically $80K per license, and the non-recurring engineering costs (NRE) for a single 0.25 micron mask set are typically $250K. Unless the production volume can justify these costs, SOC design is not feasible for most customers today. Atmel has addressed these issues and solved these problems by developing a programmable system-on-chip and a very low cost set of design tools, based on proven, industry standard software.

The concept of programmable system level integration has the potential to bring many benefits to the system designer – having the right tools will ultimately determine the success of this product area. Design engineers have made significant investments in both tool acquisition and the time required becoming an expert at using these tools. In order for FPSLIC to succeed it is mandatory that the tools used in FPSLIC design take advantage of designers' knowledge of existing tools design flows and methodology. This is true for the FPGA design, microcontroller design and the interaction between the two.

The FPSLIC software system architecture is shown in Figure 10. The design manager seamlessly integrates all the design tools, databases and flows required for FPSLIC implementation as shown in Figure 11. The design manager steps the user through each stage of the design process, from Verilog/VHDL design entry for the hardware (FPGA), through synthesis, place & route, and simulation, as well as guiding the user through the microcontroller design process and verification, as well as co-simulation and verification between the software and hardware aspects of the system.

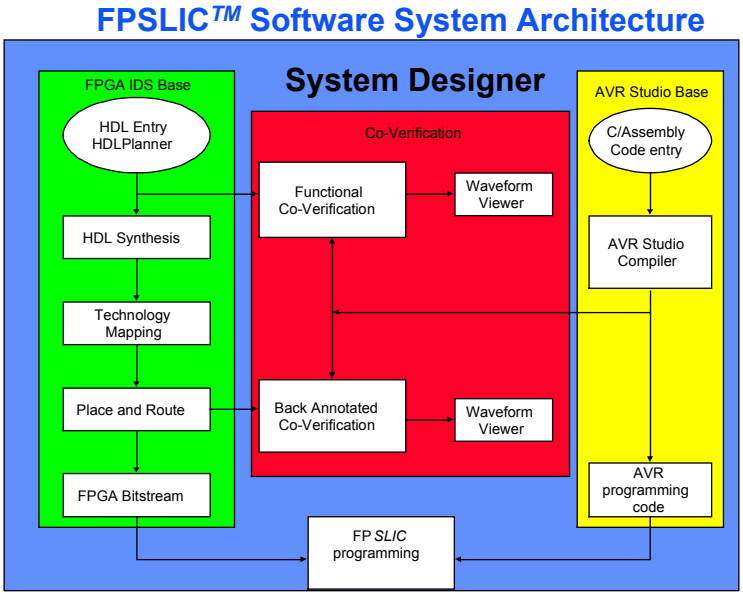


Figure 10

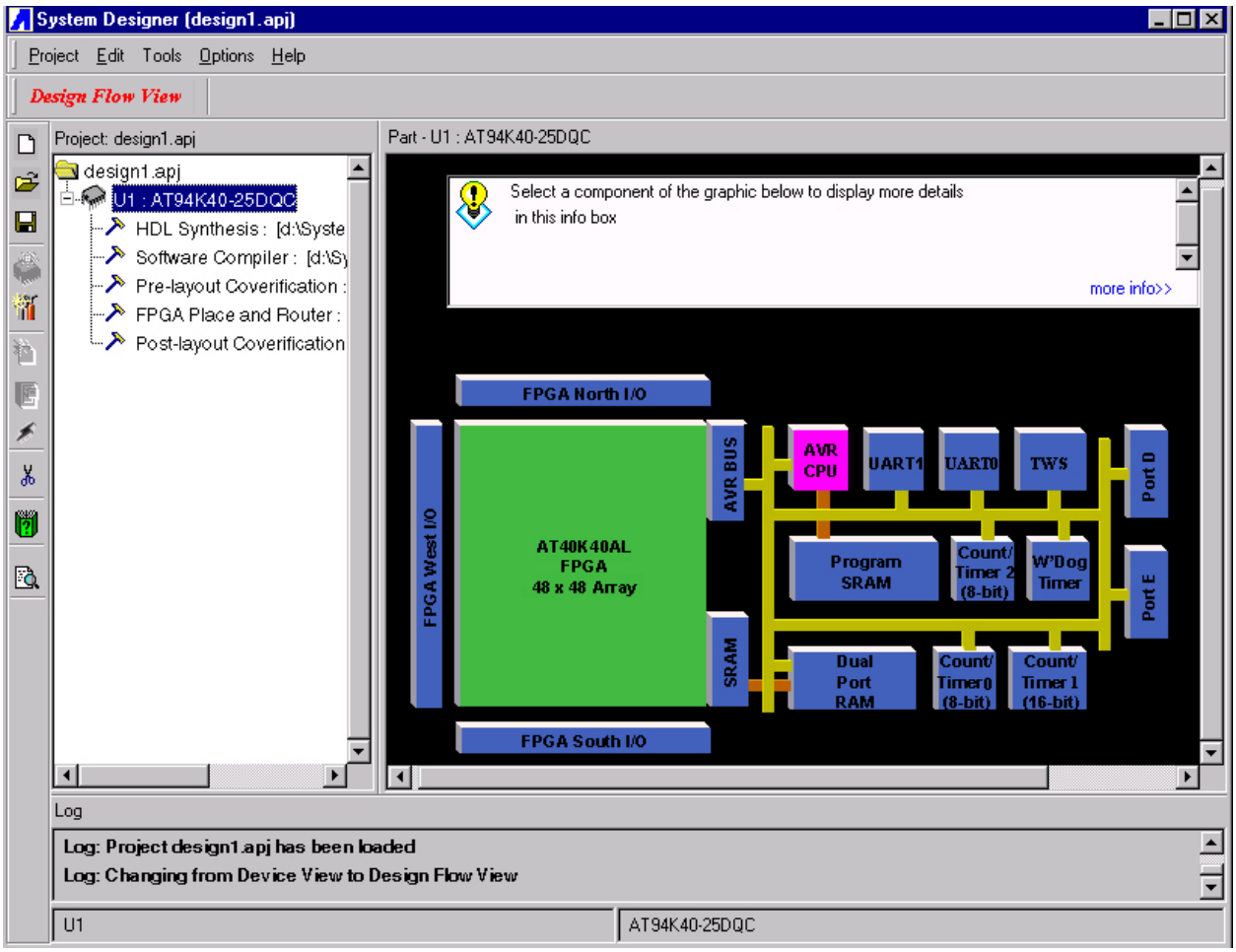


Figure 11

Included in the FPSLIC design suite are the AT40K series FPGA design software, Exemplar synthesis compiler, Model Technology hardware simulator, AVR studio and Seamless co-verification tools, powered by Mentor Graphics. Figure 12 provides additional details of the included tools. System designer has been architected in a modular and open architecture fashion. This has the added benefit of allowing the FPSLIC designer to work with other 3rd part design entry, synthesis and simulation tools. In addition, the tools can be adapted to work with addional microcontroller and processor cores, as well as different variations of the FPSLIC architecture. This powerful combination of integrated tools enable fast, efficient, simplified design for FPSLIC devices.

FPSLIC™ Design Suite

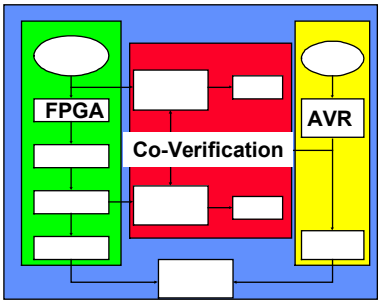
Complete IDS FPGA Software
- Place & route, floorplanning, timing analysis, etc.

- **Exemplar Synthesis Compiler**
  - VHDL & Verilog entry *(FPSLIC version)*

- **Model Technology hardware simulator *(FPSLIC version)***

- **AVR Studio**
  - Design & debugging
  - Instruction set simulator

- **Seamless co-verification tools**
  - PC-based: Windows 95/98/NT/2000
  - Powered by Mentor Graphics *(FPSLIC version)*



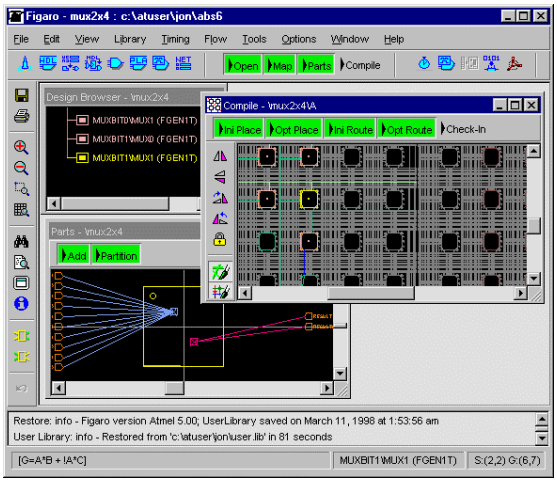
Everything* for $995

* = Does not include C compiler

Figure 12

The FPGA portion of the FPSLIC design tools are based on those developed for Atmel's industry standard, proven AT40K series FPGA. The FPGA development software (Figure 13) includes both push-button and interactive place and route, the ability to import files from other FPGA designs, logic and RAM compilers, floor planning, timing driven design, static timing analysis, incremental design change, as well as synthesis and simulation.

FPGA Development Software



FPGA Development Tools

- **Push Button 85%+ APR**
- **Multi-Chip Partitioning**
- **XNF/EDF/WIR Import**
- **Hierarchy Browser**
- **Floor planner**
- **Timing Driven Design**
- **Static Timing Analysis**
- **Bitstream Utilities**
- **Incremental Design Change**
- **Architecture Mapping**
- **Back Annotation support**
- **Extensive interactive help.**
- **Interactive Timing Analysis**
- **Interactive Layout Editor**
- **Library Manager**
- **Design re-use**

Figure 13

Co-Verification

One of the challenges of system-on-chip design is the opportunity to develop both hardware and software concurrently. With the FPGA/logic and microprocessor both on the same chip, it is only natural that co-verification tools be available to support the ability to make hardware/software tradeoffs and to verify the interaction of these prior to building the actual system board.

The conventional way of designing a system is to first design the hardware and implement it in an FPGA or ASIC, and then turn the design over to the software developers who will implement the processor code. The limitations of the traditional design approach without co-verification are shown in figure 14. The problem with designing a system where hardware is developed prior to software is that inability to anticipate and accommodate changes to the hardware do to software limitations, and visa versa. This creates a situation where numerous design iterations may occur due to performance, power and space limitations of both the logic and the processor. The situation can result in significant delays to development schedule, typically 1-3 months, increased cost due to printed circuit board revisions, and a lower performance, higher power consumption design.

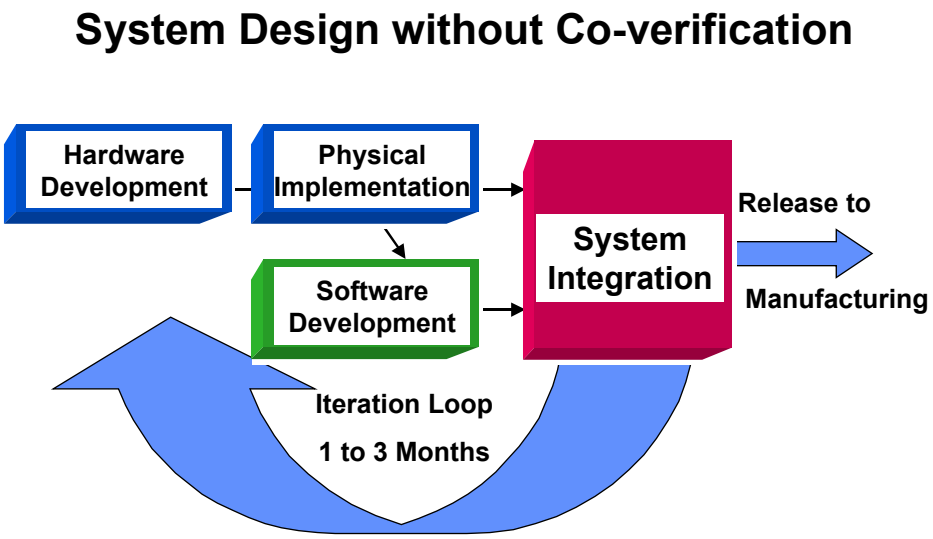


Figure 14

In order to speed up the development time, and optimize the hardware and software, the system designer should be able to develop both the hardware and software concurrently, with the ability to evaluate the impact of each on the overall system. Figure 15 represents the impact on the system design cycle by using co-verification design methodology. FPSLIC System Designer incorporates a co-verification back plane, effectively connecting the FPGA hardware simulator to the microcontroller instruction set simulator, as shown in figure 16. The use of co-verification effectively eliminates the need for an in-circuit emulator (ICE) by facilitating the verification process. This is possible by allowing the designer to make changes to the hardware/software and seeing the resultant interaction and effect on the software/hardware. This is all done on a desktop PC operating under Windows or NT.

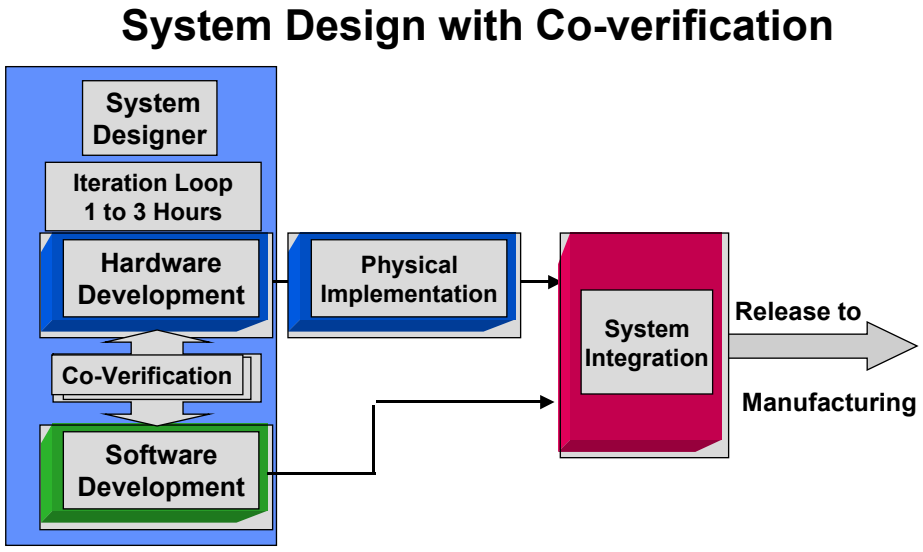


Figure 15

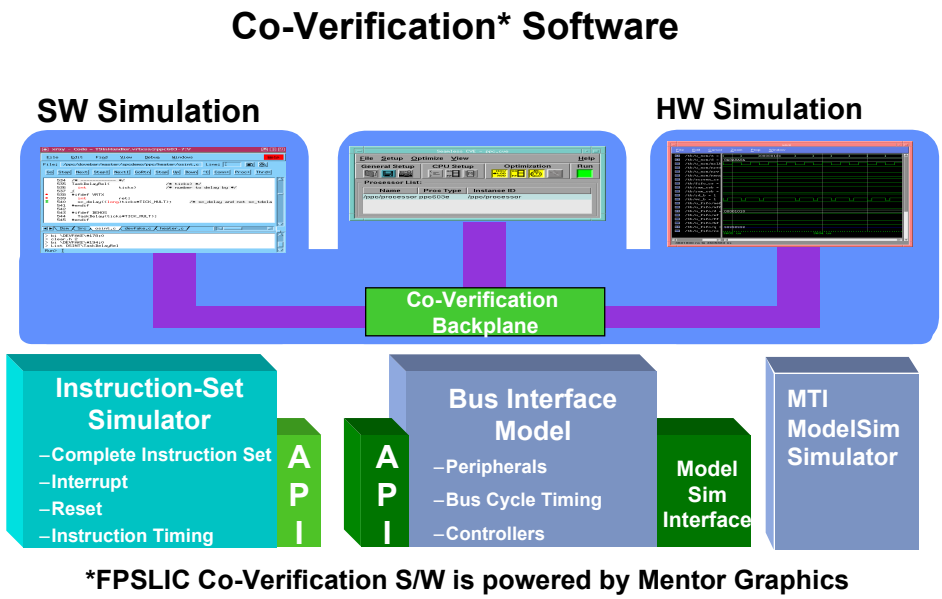


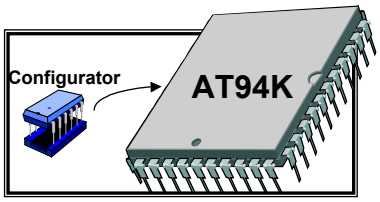
Figure 16

FPSLIC Configuration

The FPSLIC family of devices is designed to be implemented on a 0.35u and below SRAM/ASIC process. The entire FPSLIC device is SRAM-based, which brings the benefit of unlimited reconfiguration and update capability – for both the FPGA hardware and AVR microcontroller software. This means that design (hardware and software) changes can be implemented even after the system has shipped to the end-customer by using remote updates. Also, as mentioned previously, the AVR microcontroller can actually reprogram all, or parts of the FPGA – in system, on-the-fly, in real time. This allows adaptive hardware and hardware acceleration for DSP and other computer intensive applications. In fact the FPGA can actually be used as a reconfigurable co-processor or reconfigurable network processor for certain types of applications.

FPSLIC Configuration Statistics

Device	Configuration Bits*	Configurator*
AT94K10	423K	AT17LV512
AT94K20	524K	AT17LV010
AT94K40	809K	AT17LV010



* = Can be reduced by using bit-stream compression option

Figure 17

In order to support this programming ability, it is necessary to store the FPSLIC configuration data (instruction set for the AVR and FPGA) in a non-volatile external configuration memory. The data can be serially loaded into the FPSLIC device at power-up, as well as any other time, simply by downloading the data from the memory into the FPSLIC device. The data can be loaded via an independent microcontroller or directly from an Atmel AT17C/LV series Configurator. The Configurator is a family of devices that range from 65K-2M of Serial EEPROMs, all of which contain all the handshaking logic required to program SRAM-based FPGA and FPSLIC devices. Because the Atmel Configurator family are EEPROM-based, not only is the FPSLIC capable of being reprogrammed in-system, but so is the Configuration EEPROM itself. This capability makes the FPSLIC an ideal device for emerging internet appliances and "subscription" products. The configuration requirements for FPSLIC are shown in figure 17.

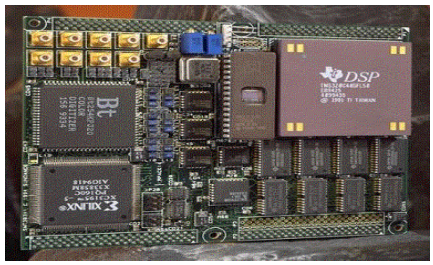
Summary

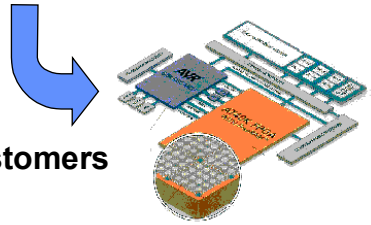
The FPSLIC family of products represents the next step in programmable design. FPSLIC essentially brings to System-on-Chip design what the FPGA brought to the Gate Array market 15 years ago. Because FPSLIC is a standard, off-the-shelf product, there are no NRE charges associated with it. A customer can purchase a single device or a million; there are no minimum volume requirements such as those associated with a customer ASIC or CBIC (cell based IC). The annual subscription cost for FPSLIC design tools are less than $1000, essentially removing the tool cost barrier.

The FPSLIC combination of architecture and design tools include co-verification, enabling concurrent software and hardware development. Important speed, power and density tradeoffs can be made early in the design process, significantly reducing development time and cost. Because FPSLIC is SRAM-based, it is infinitely reconfigurable, significantly reducing design risk by allowing both software and hardware changes to be made late in the design cycle, even in the field after shipment to the customer.

FPSLIC™ **Features/Benefits**
Summary

- **$0 NRE**
- **No minimum volume requirements**
- **Very low cost design tools**
- **Standard product**
- **Fast time-to-market**
- **Programmable design reduces design risk**
- **IP issues eliminated**





>> System level integration for all customers

Figure 18

Another advantage of FPSLIC is the elimination of intellectual property issues relating to the processor, memory and FPGA. All these functions are seamlessly integrated into the FPSLIC device, further reducing design time and cost. Additional peripheral and logic can be created and programmed into the FPGA, allowing product customization and differentiation.

The combination of features listed herein and the low cost of entry for design tools for FPSLIC result in many benefits shown in figure 18. These include reduced board area, high speed, low power and low cost.