Taking Advantage of Reconfigurable Logic

Bradly K. Fawcett
Xilinx Inc.
2100 Logic Drive
San Jose, CA 95124

ABSTRACT

Logic implemented in an SRAM-based FPGA is reconfigurable; that is, changes can be made to the system's logic functions by reprogramming the FPGA(s) in the system. Examples are cited of systems that make use of this in-system reconfigurability. These applications can be divided into three main categories based on how the FPGA's reconfigurability is applied: systems with built-in diagnostics, adaptable system designs, and systems with multi-purpose hardware.

INTRODUCTION

By combining the density and flexibility of gate arrays with the ease-of-use and convenience of a user-programmable device, Field Programmable Gate Arrays (FPGAs) have emerged as a leading application-specific IC (ASIC) technology. FPGA technology has proven to be a cost-effective alternative to standard SSI/MSI devices or custom VLSI components in thousands of system designs. However, the capabilities of static-memory-based FPGAs extend well beyond the direct replacement of other types of logic devices. With most IC technologies, hardware is indeed hard; once a given logic function is implemented in hardware, changing that logic is difficult, requiring the modification of printed circuit board traces, the addition or replacement of components, and other costly measures. However, with SRAM-based FPGAs, changes can be made to a system's logic functions simply by reconfiguring the FPGAs in the system. Innovative system designers are taking advantage of this "in-system reconfigurability" of SRAM-based FPGAs; their systems include multiple configuration programs for the FPGAs, allowing varying operations to be performed with a minimal amount of hardware. In general, these applications of reconfigurable logic fall into three categories: systems with built-in diagnostic or test logic, "adaptable" system designs, and systems with "multi-purpose" hardware.

THE MECHANICS OF RECONFIGURABILITY

FPGAs based on static memory (SRAM) technology are available from several manufacturers, including Xilinx, AT&T Microelectronics, Altera, Atmel, and Motorola. The architectural implementation varies among the manufacturers, but the basic concepts are the same. In these FPGAs, logic functions and interconnections are determined by configuration program data stored in internal static memory cells. In Xilinx devices, for example, a single-ended five transistor memory cell is used to minimize memory area. Rather than the traditional array structure, the memory cells are distributed throughout the device; each cell is placed as close as possible to the logic it controls, resulting in maximum layout efficiency. The benefits of an SRAM-based FPGA include high density, high performance, testability, manufacturability, and the flexibility inherent to a device that can be programmed while resident in the system.

Since the devices are static memory-based, a configuration program must be loaded into each FPGA when the system is powered-up. Thus, the configuration program (or programs) must reside in the system external to the FPGAs (for example, in EPROMs or on a disk). Xilinx FPGA devices feature several available loading modes to accommodate various system requirements. The choice of configuration mode is determined by the input levels of dedicated control pins on the package. These
configuration loading modes fall into two main categories: automatic modes and peripheral modes. Automatic modes (Figure 1) use on-chip control logic to load the FPGA directly from an external memory device (typically an EPROM). Both parallel and serial modes are available. In the parallel mode, the FPGA generates the address and control signals needed to read data one byte at a time from a byte-wide memory. The automatic serial mode involves a simple three-wire interface to a serial memory device. With the peripheral loading modes (Figure 2), configuration programs are written to the FPGA.

![Master Parallel Mode](image1)

**Figure 1:** The master configuration modes allow for automatic initialization of an FPGA from external memory.

![Master Serial Mode](image2)

![Parallel Peripheral Mode](image3)

**Figure 2:** In the peripheral loading modes, the FPGA is programmed like a peripheral device.

![Serial Peripheral Mode (Slave mode)](image4)

![State diagram of the FPGA configuration process](image5)

**Figure 3:** State diagram of the FPGA configuration process.
under the control of an external processor or DMA device, just like any other peripheral device. In the parallel peripheral mode, a byte of data is written to the FPGA during each write cycle. The serial peripheral mode is a simple two-wire interface, where data are written to the FPGA at a rate of one bit per clock period.

Just as these devices are initially configured during system initialization, they can be re-configured at any time during system operation. A reconfiguration of a Xilinx FPGA is triggered by a high-to-low transition on the dedicated PROGRAM input. Figure 3 shows a state diagram of the FPGA configuration process. During initial power-up, an internal timer imposes a delay to allow the power supply voltage to stabilize; this step does not occur when an already configured part is being reconfigured. Next, all configuration memory cells are cleared. An open-drain status output, INIT, indicates when the initialization and memory clear operations are completed. Then, the RESET input is sampled, and, if inactive, the mode pins are sampled and the configuration process is initiated. When the configuration memory is full, a synchronous start-up sequence is executed, and the FPGA becomes operational. All configuration memory cells are cleared and a complete configuration program is loaded each time the device is configured. Configuration loading time usually is not a performance issue, since it has no impact once the configuration memory has been programmed. Nonetheless, configuration times are short—typically just a few milliseconds. The parallel configuration modes do not hold any performance advantage over the serial modes, since all configuration data is serialized within the FPGA. Table 1 shows configuration program sizes and times for a sampling of Xilinx FPGA devices. There is no limit to the number of times an SRAM-based FPGA can be programmed.

<table>
<thead>
<tr>
<th>Table 1: Configuration program size and time for selected FPGAs (serial peripheral mode, maximum configuration clock rate)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Device</strong></td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>XC3020</td>
</tr>
<tr>
<td>XC3042</td>
</tr>
<tr>
<td>XC3090</td>
</tr>
<tr>
<td>XC4002A</td>
</tr>
<tr>
<td>XC4006</td>
</tr>
<tr>
<td>XC4013</td>
</tr>
</tbody>
</table>

The design-related benefits of SRAM-based programmable logic are somewhat obvious. The ability to reconfigure FPGAs resident in the target system significantly eases the debugging process, reducing overall development time and cost, and shortening the product's time-to-market. A download cable provided with the development system allows configuration programs to be downloaded directly from a personal computer or workstation to an FPGA device resident in the target system. Thus, designers can immediately check the results of design changes in the target system. Temporary modifications to the logic, such as routing an internal node to an unused I/O pad, can be implemented for debugging purposes and then removed from the production design. Typically, small design changes can be implemented in a few minutes time, with the designer receiving almost immediate feedback on the effects of the design modifications. There is no lengthy wait for a custom device to be manufactured, and no wasted components as with one-time-programmable devices; there is not even the inconvenience of removing the part from the target system for erasure, as with EPROM-based devices. The flexibility of SRAM-based FPGAs greatly reduces design risks and encourages experimentation during the design cycle. In essence, SRAM-based FPGAs provide a flexible means of 'breadboarding' logic designs, as well as a cost-effective means of implementing logic in the final product.

However, the benefits of the reconfigurable nature of SRAM-based FPGAs can extend beyond the debugging stage and influence the nature of the final product itself. Systems can include multiple configuration programs for their FPGAs, allowing varying operations to be performed efficiently with a
minimal amount of hardware. Reconfigurable logic has been used to implement system diagnostics, to create systems that are adaptable for different environments or operations, and to implement 'multi-purpose' hardware in a system. The result is smaller, more powerful, less expensive, and more reliable system designs.

RECONFIGURING FOR SYSTEM DIAGNOSTICS

Built-in diagnostic and test logic assists the designer during initial system debug, increases the confidence level of the system's users, and aids in field maintenance operations. However, using traditional logic technologies, the additional logic needed to implement diagnostic operations can significantly increase the system's size, complexity, and cost.

Reprogrammable SRAM-based FPGAs offer the designer an economical means to implement test logic without adding to the hardware content of the system. The same FPGAs that implement the system's logic can be reconfigured while in the system to hold diagnostic logic to test that system. When the system is powered-up or placed in a test mode, its FPGAs are configured with logic functions dedicated to testing other circuitry in the system. For example, a configuration dedicated to implementing a boundary scan test of the board assembly could be loaded into the system's FPGAs.1 (Boundary scan test logic is built into the XC4000 family of FPGAs, but would need to be implemented as a separate configuration of an XC2000 or XC3000 family device.) Once the testing is successfully completed, other configuration programs are loaded into the FPGAs to implement the actual logic of the particular end application intended for that system. The diagnostic test logic is essentially 'cost-free', except for the additional memory space required for the extra FPGA configuration programs.

For example, designers at telecommunications equipment supplier Tellabs Inc. (Lisle, IL) used this strategy in a voice compression module for a T1 multiplexer (Figure 4). The design includes two XC2018 FPGAs. During normal operation, one FPGA provides all the interface logic for the board's microcontroller, RAM, and system backplane, arbitrating accesses to the RAM from the controller and the main system. The second FPGA contains most of the 'glue logic' for the data compression operation. However, both FPGAs can be loaded with special diagnostic configurations. In the test mode, the first FPGA connects the microcontroller to the RAM for memory testing, and monitors controls on the system's backplane. The second FPGA can receive timing information from the microcontroller instead of the system backplane, verify the data paths, and check the contents of the EPROM used to implement the code converter's companding algorithm. Actually, two different test configurations have been generated, and others are planned for future product upgrades. All the configuration data is present in memory on the board; the microcontroller handles the downloading of FPGA configuration programs. Similar schemes have been employed to include diagnostic logic in applications such as computer peripherals, industrial controllers, medical equipment, and IC testers.2-5

ADAPTABLE SYSTEM DESIGNS

A popular use of reconfigurable logic is the implementation of a single hardware design that can be adapted for varying tasks or environments. In such systems, a number of potential configuration programs can be downloaded into the system's FPGAs to alter the logic for particular applications or operations as needed. Hence, more functions are implemented with fewer components, hardware design costs can be amortized over a greater number of systems, and design cycle times are greatly reduced. Often, diagnostic configurations, as described above, are included as one of the configuration options.

Multiple potential FPGA configuration programs can be included in a system, and the appropriate configuration program downloaded dependent on the user's selection of the system's operating mode. This strategy was employed in a network controller designed by Perception Inc. (Prior Lake, MN) and manufactured by Computer System Products (Minneapolis, MN). The SMARTStar Active Repeater allows users of midrange IBM AS400 and System 3x series computers to replace twinax daisy chain network configurations with a star-wired, twisted-pair-based network. Network control logic implemented in an XC3000 family FPGA can be configured in either of two modes, as determined by a slide switch on the back of the unit. With the exception of the analog interface to the network cable, all
the logic of the repeater is implemented in the FPGA, including error detection, signal distribution, direction sensing, timing and synchronization, and diagnostic logic. The choice of modes is dependent on how the user wants corrupted messages to be handled. In the "normal" mode, where clean signals are expected, the repeater examines the data stream and discards anything that does not conform to the IBM standard, assuming that it is noise. If the data stream is accepted, it is stored, a new header is constructed, and the message is transmitted. If actual messages are being corrupted by noise and discarded by the repeater, the transmitting node will never receive an acknowledgment, generating an error message at the transmitting node, and, thereby, notifying the system administrator that a problem exists. In the "noisy" mode, the repeater will transmit messages even if they are corrupted; the receiving node will also detect the error, and acknowledge with a return message requesting a retransmission. This prevents system errors, but potentially lowers the effective throughput of the network.

TDX Peripherals Inc. (Hauppauge, NY) took advantage of 'adaptable hardware' in the design of a PC-compatible controller card in the TDX 50/GCR 9-track, 1/2-inch tape drive. Four different data densities commonly are used for 9-track tapes, ranging from 800 to 6250 bits per inch (bpi). Each density level has a different data encoding scheme, with the complexity of the encoding algorithm increasing with higher density. TDX's former systems required a different controller card design for each. By employing seven FPGA devices, including four XC3000 series and three XC2000 series parts, a single PC card capable of handling several different densities and formats was developed. Customers can select the desired format from a menu on the PC's terminal; this selection triggers the downloading of the appropriate configuration programs and software device driver. All seven FPGAs
are downloaded in parallel from the PC's disk under control of the processor, and constitute over 70% of the logic in the system. Initial shipments of the system supported the 1600 and 6250 bpi formats. The lesser used 3200 and 800 bpi formats were added later through the development of new FPGA configuration programs, and previous customers' systems were updated via floppy disk.

Such 'adaptable system' design has been particularly popular among test equipment manufacturers; reconfigurable FPGAs can be used to adapt the same hardware to perform varying types of tests. This strategy was incorporated in the design of the Innovage 2000 microprocessor system troubleshooter (Innovage Technologies Inc., Calgary, Alberta, Canada). Used in conjunction with a host PC or workstation, the Innovage 2000 performs functional tests of microprocessor-based boards using DMA emulation; the tester connects to the microprocessor on the board and behaves like a DMA controller to access the board's circuitry. Boards also can be checked dynamically while the processor is running using cycle-stealing techniques. The tester's interface to the microprocessor is implemented in multiple XC2000 and XC3000 FPGA devices, allowing for easy modifications to support several different 16-bit and 32-bit microprocessors. Users can purchase interface pods and software packages, including the appropriate FPGA configuration programs, that personalize the tester for a particular processor type. Currently, support is available for several Intel and Motorola processors, including the 80486. Easy field upgrades are another bonus provided by the use of reconfigurable FPGAs; the Innovage 2000 includes a modem port that can be used to download new FPGA configuration programs and other software updates to upgrade or modify units in the field.

Dainippon Screen Manufacturing Co. (Kyoto, Japan) used SRAM-based FPGAs in a system that automates optical inspections of the metal pattern on printed circuit boards (Figure 5).[6] Four XC2018 FPGAs are used, with each controlling the testing of one of four design rule checks: line width, clearance width, spurious copper size, and pinhole size. Image data from a CCD camera is loaded into the XC2018 devices, converted to a matrix format, and processed for defect correction. Defective block data is passed to the system's processor through a FIFO buffer. There are eleven levels of criteria for line and clearance width, and five levels of criteria for pinhole and spurious copper size. For any given

![Diagram](image)

Figure 5: Block diagram of design rule checking logic in printed circuit board optical inspection unit
inspection run, the operator chooses the appropriate inspection criteria based on the customer's requirements; the system's processor then downloads the appropriate FPGA configuration programs corresponding to the operator's selections.

MULTI-PURPOSE HARDWARE

Taking the concept of adaptable hardware one-step further, some innovative designers have built systems where reconfiguration of the programmable logic is a normal part of the system's operation. This implies that the system has some "mutually-time-exclusive" functions that can be swapped into and out of the FPGAs as needed. For example, at any given time, a tape recorder can read or write, but never both simultaneously. Consequently, an FPGA within a digital tape recorder could be configured to perform one set of functions when writing data, and then reprogrammed to perform another set of functions when reading data. These types of "multi-purpose" hardware applications of SRAM-based FPGAs are especially cost-effective; at least twice the hardware would be required to implement the same functionality with traditional logic devices.

This strategy was employed by Tektronix Inc. (Wilsonville, OR) in the design of the PhaserCard printer controller. This PC-bus compatible card uses an XC2018 FPGA to implement the printer interface logic (Figure 6). Included in the FPGA is a state machine to control the flow of data from an external FIFO buffer to the printer, control and status registers accessible by the card's microprocessor, data formatting logic, handshake control logic, timing signal generators, and interrupt control logic. Two separate printer interface ports can be controlled by the FPGA, allowing two printers to be connected simultaneously. By loading the appropriate configuration program into the FPGA, the PhaserCard can be used as a controller for any of several printers, including a monochrome laser printer, ink-jet color printer, and wax-transfer color printer, all with parallel interfaces, and a wax-transfer color printer with a synchronous serial interface. When two printers are connected to the card, selection of the correct configuration program is controlled by a microprocessor; when a printer is to be accessed, the FPGA is automatically configured for that particular interface. The FPGA is reconfigured frequently during idle (non-printing) periods to check each printer's status. A special diagnostic configuration also was included in the production design to facilitate field servicing.

Reconfigurable SRAM-based FPGAs were key to the design of a pivoting monitor from Radius Inc. (San Jose, CA), that can be used in either portrait or landscape orientation. The monitor

![Figure 6: Block diagram of PhaserCard printer controller](image)
actually has six potential operating modes: 1, 2, and 4 bits/pixel for the various shades of gray in portrait mode, and 1, 2, and 4 bits/pixel in landscape mode. The interface control circuitry, including pixel address translation, for any one mode fits in a single XC2018 FPGA. Thus, six potential configuration programs are available for the FPGA. A position sensitive switch senses when the monitor is rotated and generates an interrupt to the system software, which in turns triggers the automatic downloading of the appropriate FPGA configuration data.

As well as having different configuration programs supplied to adapt to different processors, the Innovage 2000 microprocessor system tester described earlier also uses "on-the-fly" reconfigurability. Each microprocessor interface has to support a generic list of test-oriented commands, such as Bus Test, Memory Test, Breakpoint, and Trace. Commands can be selected from a menu individually, or appended together to be stored and executed as a multi-command test routine. Depending on the type of test selected for execution, the appropriate configuration programs are downloaded to the FPGAs by the processor in the host PC or workstation, allowing many types of tests to be implemented with a minimal amount of hardware. The configuration files are stored in flash memory devices located with the FPGAs on the interface board, and their downloading is transparent to the test instrument user.

Using in-system reconfigurability to minimize the amount of required hardware is particularly attractive in portable test equipment applications like the Innovage 2000, where size, weight and cost are primary considerations. These factors led to the use of FPGAs in the appropriately-named Chameleon 5-BRI/PRI portable protocol analyzer from Tekelec (Calabasas, CA). The Chameleon 5 is a multifunction field service unit for complete Basic Rate ISDN, Primary Rate ISDN, T1, and wide area network testing. Its hardware includes an XC3042 FPGA that is dynamically loaded from one of several different configuration files under the control of the unit's microprocessor, dependent on the task being performed. For example, the Chameleon 5 can be used to generate two different types of test patterns during T1 burst error rate testing (BERT): fixed patterns and pseudo-random patterns. Figure 7 illustrates how the XC3042 FPGA is connected to the unit's two T1 interfaces; all the T1 channels' clock and data signals are available to the FPGA. An external multiplexer controls the outgoing data's source. The fixed pattern generator consists of a set of registers that store the data pattern, while pseudo-random patterns are generated using a variable-length shift register feedback ring. In order to implement both (along with transmit and receive multiplexers, configuration and error registers, and additional support logic), the XC3042 FPGA is simply reloaded with the appropriate configuration whenever the test pattern type needs to change.

![Diagram](image-url)

Figure 7: The FPGA has access to all the clock and data lines for two T1 channels in a telecommunications tester
Tekelec took further advantage of reconfigurable logic to update units already in the field and avoid product obsolescence. Fractional T1 networks were not in use when the Chameleon 5 was original conceived and designed, but fractional T1 BERT testing has since become a requirement for T1 field test equipment. Normally, adding new capabilities would require a hardware redesign. However, Tekelec engineers were able to produce an FPGA configuration to support fractional T1 capability without altering the tester’s hardware, and avoided any impact on their board manufacturing operation or inventories. A software update distributed on floppy disks allowed previous customers to upgrade to this new capability.

For the same reasons, Phoenix Microsystems (Huntsville, AL) chose the XC4006 FPGA as the main logic element in its latest telecommunications test unit. The 5565/96B unit performs drop and insert DS0 level testing, in addition to a full range of SLC-96 DDL testing. All the unit's control logic is incorporated within the FPGA, including logic to drop and insert T1 data in a variety of formats. Since there are a number of independent operating modes, the XC4006 FPGA is dynamically reprogrammed to implement the desired functionality. The various FPGA configuration programs are stored in flash PROMs accessible via an external port on the tester, facilitating future upgrades of units in the field.

Massively Reconfigurable Systems

Some applications take the concept of multi-purpose, reconfigurable logic to the extreme, with entire boards full of FPGAs that can be configured with any desired logic functions.

Several vendors, including Quickturn Design Systems, Zycad, and Aptix, offer ASIC emulation systems based on in-system programmable FPGAs. In these systems, multiple FPGAs are used to emulate logic designs comprising tens of thousands of gates. Software in these systems accepts a netlist describing a large design targeted for a custom IC, partitions that netlist among multiple FPGA devices, places and routes the design within a matrix of FPGAs, and determines the maximum speed for correct functional operation of the resulting implementation. Thus, a designer can emulate and debug the operation of any large digital design before committing it to silicon. Such an emulation system was used during the design of the Intel Pentium microprocessor.

Reconfigurable FPGA technology is already shaping the future of computing. Research laboratories have used SRAM-based FPGAs to implement multi-purpose, high-speed coprocessors for accelerating operations in scientific computer systems. These "reconfigurable engines" can provide performance exceeding supercomputers in some applications.

For example, the SPLASH system, a linear logic area designed by the Supercomputing Research Center [9] includes a 32-stage linear logic array with a VME interface to a Sun workstation (Figure 8). Each stage consists of an XC3090 FPGA and a 128 Kbyte static memory buffer that can be used as a scratchpad during operations, preloaded with initial conditions, or read directly by the host after program execution. XC3090 FPGAs also are used to implement the VME and VSB bus interfaces. SPLASH's

![Figure 8: The SPLASH coprocessor board features a linear array of FPGA and SRAM devices](image-url)
first application was to implement a systolic algorithm for one-dimensional pattern matching during DNA research, where it outperformed a Cray-2 by a factor of 325 and a custom-built nMOS device by a factor of 45. A second-generation system, based on XC4010 FPGAs, is in development.\[10]\n
Other researchers have gained equally impressive results. Digital Equipment's Paris Research Lab has built two versions of a single-board, reconfigurable coprocessor for DEC workstations: PeRLe1, based on XC3020 FPGAs, and PeRLe2, based on XC3090 devices.\[11]\n\nThe PeRLe2's processing engine consists of a 4-by-4 matrix of XC3090 FPGAs and 32 Mbytes of RAM. Four additional XC3090 FPGAs act as cross-bar switches, and one more XC3090 device holds the bus interface logic. The PeRLe systems have been successfully applied in areas such as RSA cryptography, string matching, LaPlace equations, Newton's mechanics, neural network emulation, and graphics acceleration. IBM Almaden Research Center's GANGLION processor includes 36 XC3090 FPGAs on a single VME card, and has been used to build a neural net that executes 20 million decisions per second in a machine vision application.\[12]\n
SUMMARY

Designers are taking advantage of the reconfigurable nature of SRAM-based FPGAs in a wide variety of applications. However, these systems have a few characteristics in common. Each, obviously, has multiple FPGA configuration bitstreams available to the system in semiconductor or magnetic memory. Each has logic functions that do not need to exist simultaneously within the system. Each has a stimulus that tells the system when to change bitstreams, and enough 'intelligence' to select the correct bitstream at the appropriate time. The selection mechanism can be as simple as a switch. However, in most cases, a microprocessor or microcontroller controls the selection and downloading of FPGA bitstreams as part of its control operation.

In summary, the advent of in-system programmable, SRAM-based FPGAs has freed the designer from the "hard" nature of traditional logic ICs. With these flexible devices, designers can build cost-effective diagnostic logic, create adaptable systems, and have logic dynamically reconfigured during system operation. New system architectures that take advantage of reconfigurable logic will continue to emerge as FPGA performance and density levels continue to increase.

References