US007554358B1

(12) **United States Patent**
Fontana et al.

(10) **Patent No.: US 7,554,358 B1**
(45) **Date of Patent: Jun. 30, 2009**

(54) **PROGRAMMABLE LOGIC DEVICES WITH USER NON-VOLATILE MEMORY**

(75) Inventors: **Fabiano Fontana**, San Jose, CA (US); **Henry Law**, Los Altos, CA (US); **Howard Tang**, San Jose, CA (US); **Om P. Agrawal**, Los Altos, CA (US); **David L. Rutledge**, Hillsboro, OR (US)

(73) Assignee: **Lattice Semiconductor Corporation**, Hillsboro, OR (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 78 days.

(21) Appl. No.: **11/397,985**

(22) Filed: **Apr. 5, 2006**

(51) **Int. Cl.**
H01L 25/00 (2006.01)

(52) **U.S. Cl.** **326/41**; 326/47; 326/101

(58) **Field of Classification Search** 326/37–41, 326/47
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,548,228	A	8/1996	Madurawe	
5,640,107	A	6/1997	Kruse	
5,689,516	A	11/1997	Mack et al.	
5,696,455	A	12/1997	Madurawe	
6,049,222	A	4/2000	Lawman	
6,150,837	A	11/2000	Beal et al.	
6,172,520	B1 *	1/2001	Lawman et al.	326/38
6,304,099	B1	10/2001	Tang et al.	
6,356,107	B1	3/2002	Tang et al.	
6,467,009	B1	10/2002	Winegarden et al.	
6,538,468	B1	3/2003	Moore	
6,704,850	B1	3/2004	Reynolds	
6,714,041	B1	3/2004	Darling et al.	
6,721,840	B1	4/2004	Allegrucci	
6,732,263	B1	5/2004	May et al.	
6,774,668	B1	8/2004	Wirtz, II	
6,828,823	B1	12/2004	Tsui et al.	
6,851,047	B1	2/2005	Fox et al.	
6,873,177	B1	3/2005	Wennekamp et al.	
7,154,297	B1 *	12/2006	Camarota et al.	326/40
7,190,190	B1 *	3/2007	Camarota et al.	326/38
7,268,587	B1 *	9/2007	Pham et al.	326/41
7,346,860	B1	3/2008	LeBlanc	
2005/0189962	A1 *	9/2005	Agrawal et al.	326/41

OTHER PUBLICATIONS

U.S. Appl. No. 11/398,437, filed Apr. 5, 2006, Tang et al.
U.S. Appl. No. 10/809,658, Howard Tang.
U.S. Appl. No. 11/243,255, Howard Tang.
U.S. Appl. No. 11/293,941, Howard Tang.
U.S. Appl. No. 11/350,436, Howard Tang.
Lattice Semiconductor Corporation, ispXP™ Configuration Usage Guidelines, Technical Notes TN1026, Aug. 2002, pp. 1-18.

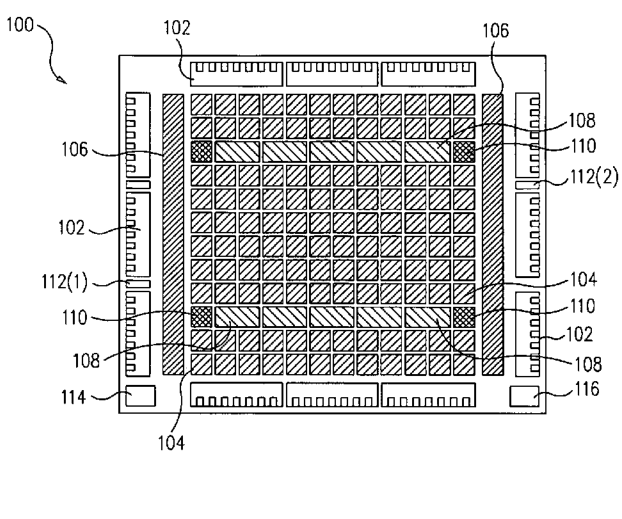
(Continued)

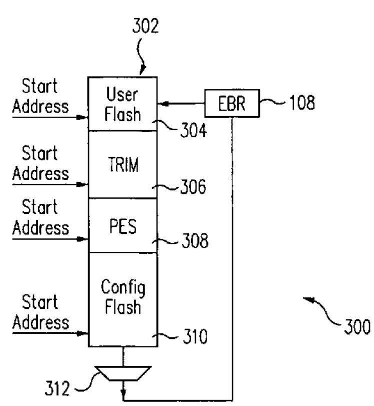
Primary Examiner—Rexford N Barnie
Assistant Examiner—Thienvu V Tran

(57) **ABSTRACT**

Systems and methods are disclosed herein to provide improved non-volatile storage techniques for programmable logic devices. For example, in accordance with an embodiment of the present invention, a programmable logic device includes a plurality of logic blocks, a plurality of input/output blocks, and a volatile memory to store data within the programmable logic device, with configuration memory adapted to store first configuration data for configuration of the logic blocks, the input/output blocks, and the volatile memory of the programmable logic device. The programmable logic device further includes a non-volatile memory adapted to store data provided from the volatile memory.

17 Claims, 4 Drawing Sheets





OTHER PUBLICATIONS

Lattice Semiconductor Corporation, ispXPGA™ Family, Preliminary Data Sheet, Sep. 2003, pp. 1-112.

Lattice Semiconductor Corporation, ispXPGA™ Family, Dec. 2002, pp. 1-90.

Lattice Semiconductor Corporation, LatticeXP Family Data Sheet, Version 01.0, Feb. 2005, 77 pages.

Altera®, MAX II Device Handbook, http://www.altera.com, M115V1-1.7, 330 pages.

Xilinx, On the Fly Reconfiguration with CoolRunner-II CPLDs, Application Note: CoolRunner-II CPLDs, May 15, 2003, pp. 1-10.

Actel Corporation, "ProASIC*mt;epmubPLUScm;0ubxmx Flash Family FPGAs*", *v.3.4 Dec.* 2003, *pp. i-iv and I-I-I-*64 & 2-1-2.67 & 2-1-2-7 (*list of changes*).

Actel Corporation, ProASICE Flash Family FPGAs, ARM7™ Soft IP Support in ProASIC3E ARM7-Ready Devices, Oct. 2005, 166 pages.

* cited by examiner

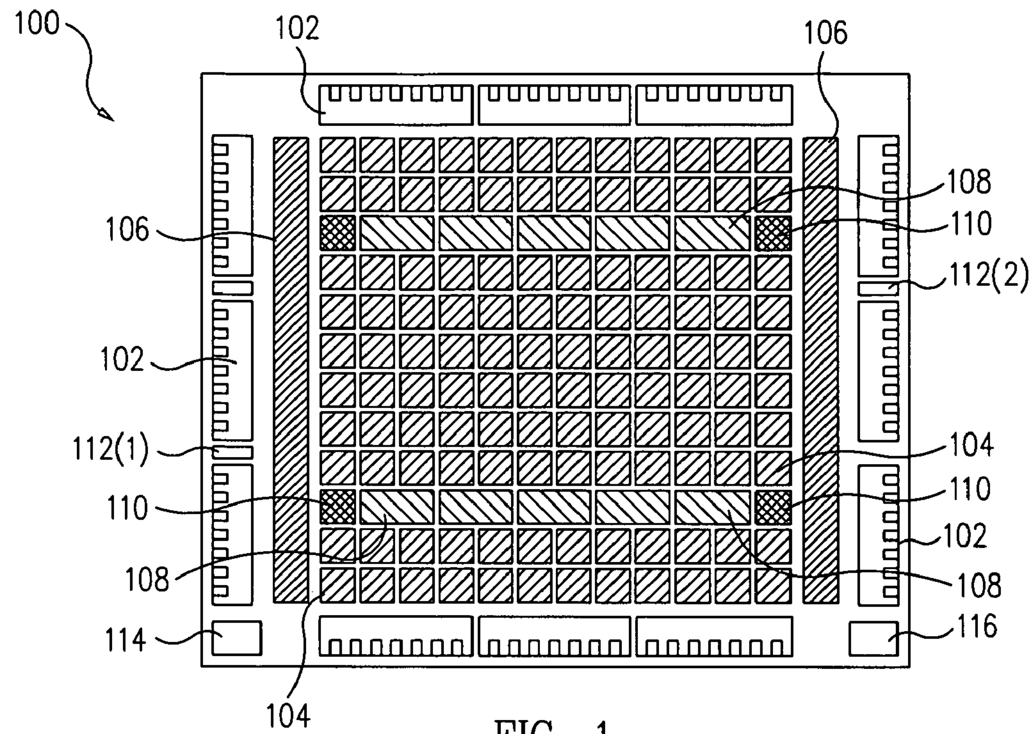

100
102
106
106
108
110
112(2)
102
112(1)
104
110
110
102
108
108
114
116
104

FIG. 1

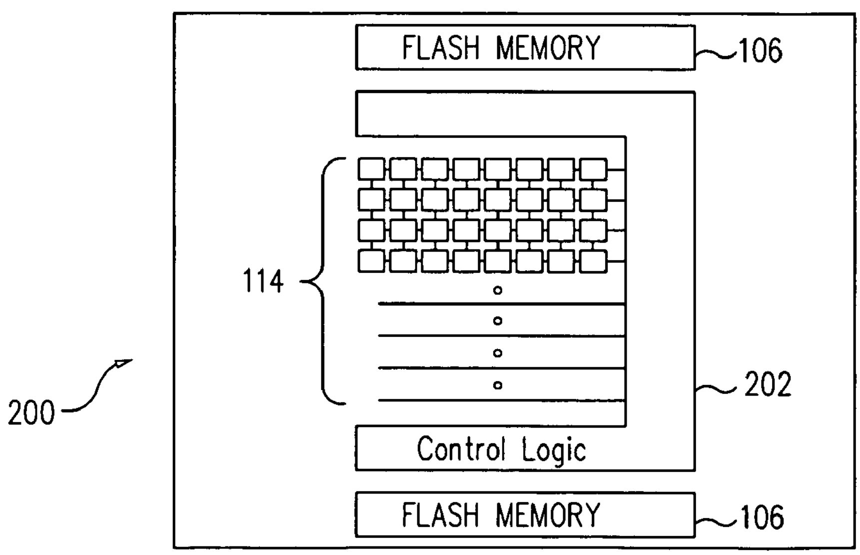

FLASH MEMORY
106
114
202
200
Control Logic
FLASH MEMORY
106

FIG. 2

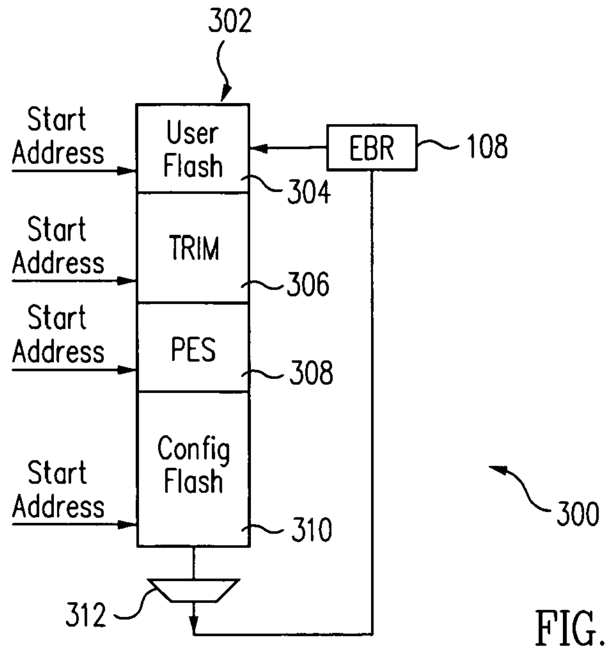

302
Start Address
User Flash
EBR
108
304
Start Address
TRIM
306
Start Address
PES
308
Config Flash
Start Address
310
300
312

FIG. 3

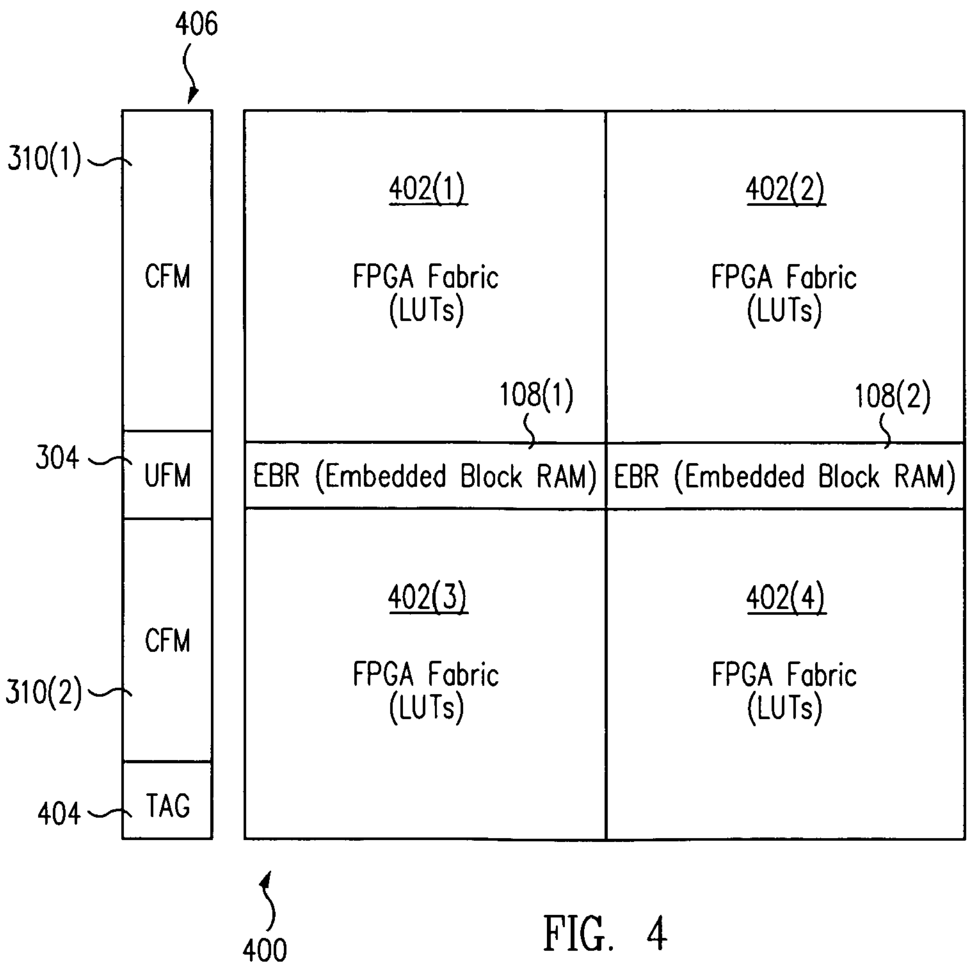

406
310(1)
CFM
402(1)
FPGA Fabric (LUTs)
402(2)
FPGA Fabric (LUTs)
108(1)
108(2)
304
UFM
EBR (Embedded Block RAM)
EBR (Embedded Block RAM)
CFM
402(3)
FPGA Fabric (LUTs)
402(4)
FPGA Fabric (LUTs)
310(2)
404
TAG
400

FIG. 4

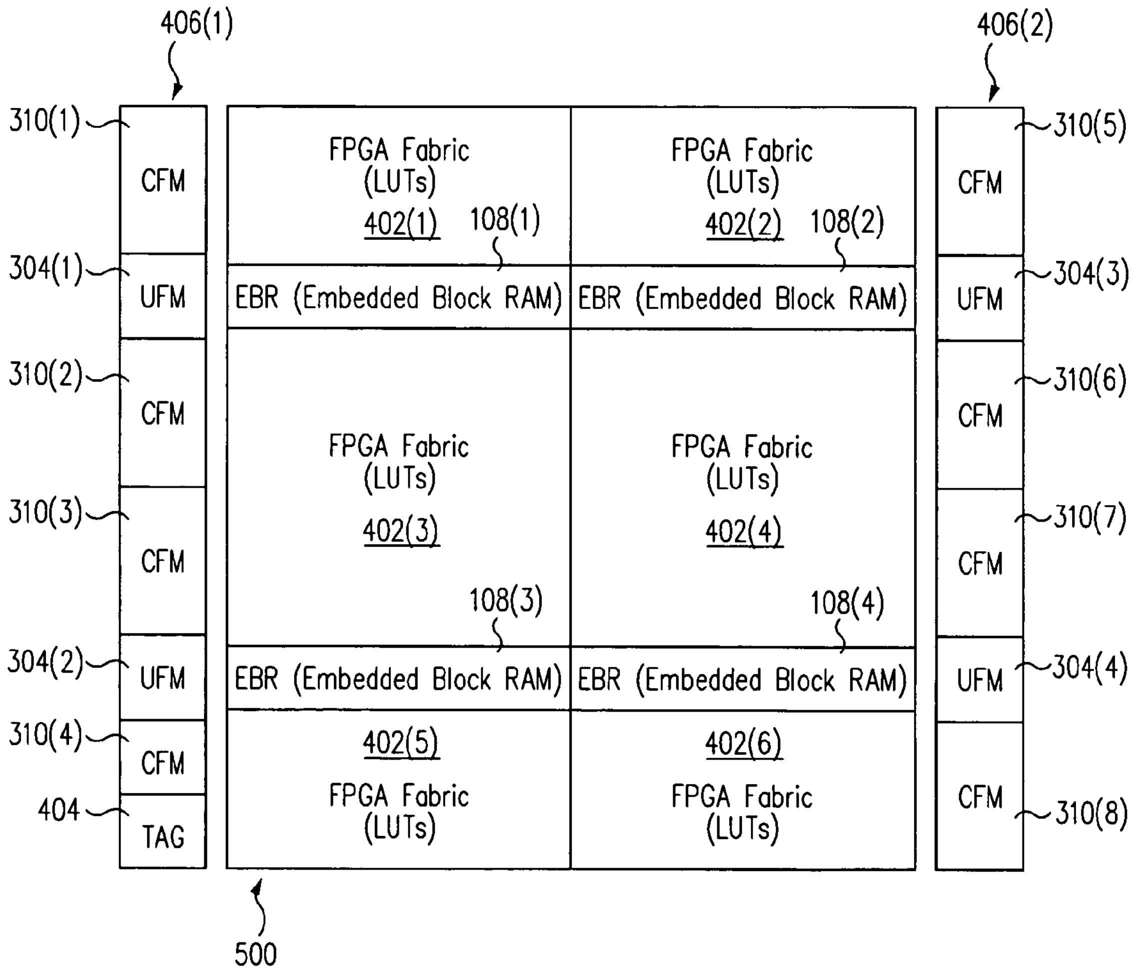

406(1)
406(2)
310(1)
CFM
FPGA Fabric (LUTs)
402(1)
108(1)
FPGA Fabric (LUTs)
402(2)
108(2)
CFM
310(5)
304(1)
UFM
EBR (Embedded Block RAM)
EBR (Embedded Block RAM)
UFM
304(3)
310(2)
CFM
FPGA Fabric (LUTs)
402(3)
FPGA Fabric (LUTs)
402(4)
CFM
310(6)
310(3)
CFM
108(3)
108(4)
CFM
310(7)
304(2)
UFM
EBR (Embedded Block RAM)
EBR (Embedded Block RAM)
UFM
304(4)
310(4)
CFM
402(5)
FPGA Fabric (LUTs)
402(6)
FPGA Fabric (LUTs)
CFM
310(8)
404
TAG
500

FIG. 5

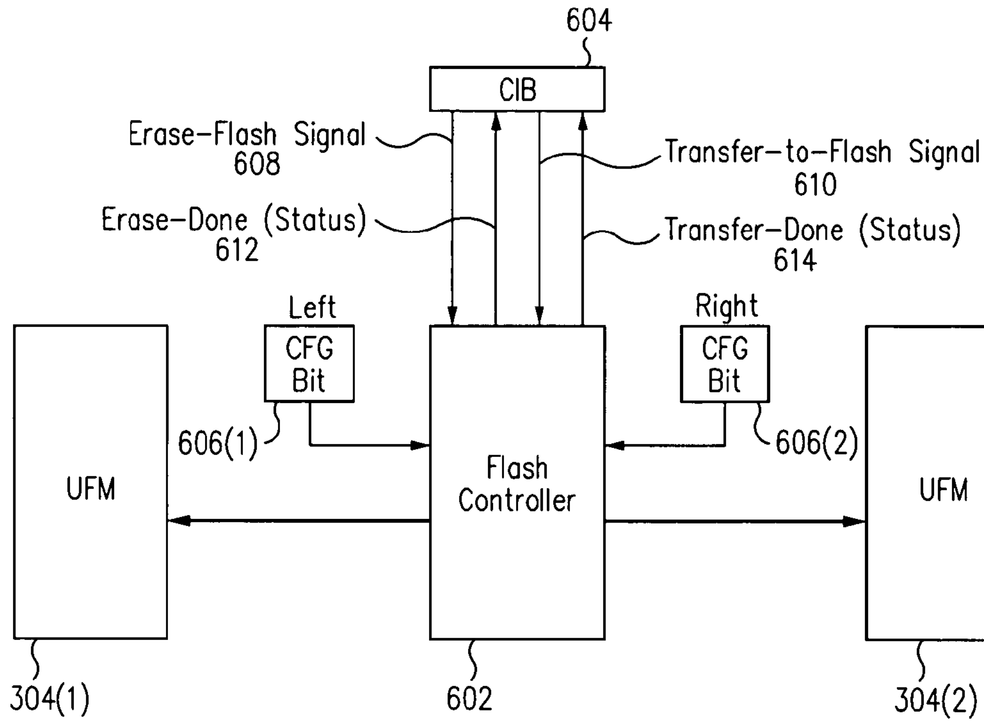

604
CIB
Erase-Flash Signal 608
Transfer-to-Flash Signal 610
Erase-Done (Status) 612
Transfer-Done (Status) 614
Left
CFG Bit
Right
CFG Bit
606(1)
606(2)
UFM
Flash Controller
UFM
304(1)
602
304(2)

FIG. 6

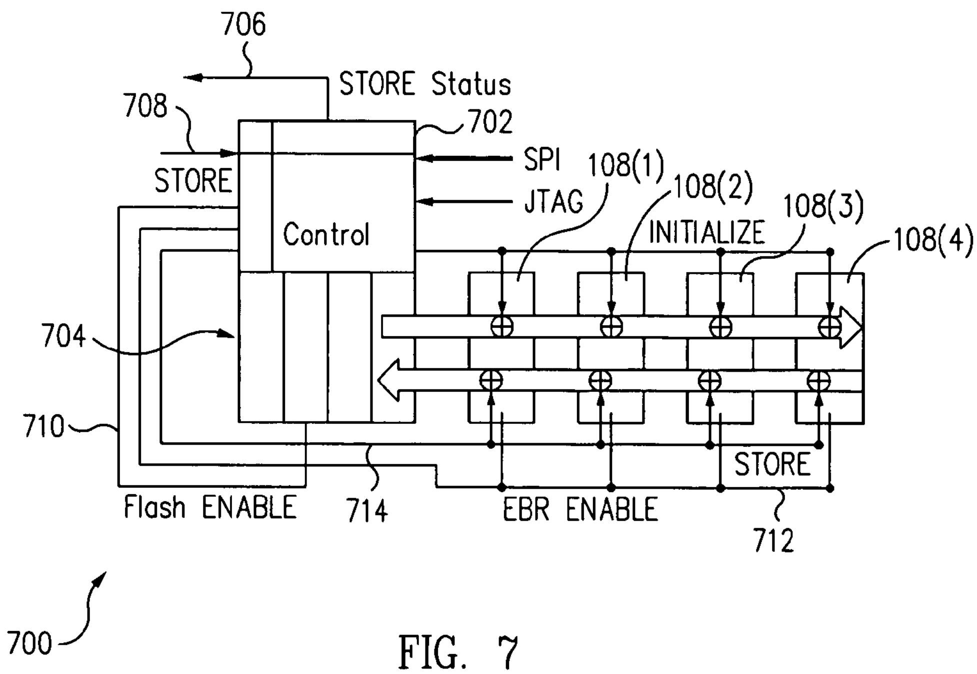

706
STORE Status
708
702
STORE
SPI
JTAG
108(1)
108(2)
108(3)
108(4)
INITIALIZE
Control
704
710
STORE
Flash ENABLE
714
EBR ENABLE
712
700

FIG. 7

PROGRAMMABLE LOGIC DEVICES WITH USER NON-VOLATILE MEMORY

TECHNICAL FIELD

The present invention relates generally to electrical circuits and, more particularly, to programmable logic devices with non-volatile memory.

BACKGROUND

A programmable logic device, such as field programmable gate array (FPGA) or a complex programmable logic device (CPLD), may be used in a variety of applications. A programmable logic device (PLD) offers the advantage of being reprogrammable in the field (e.g., while on the circuit board in its operational environment).

A drawback of a conventional PLD is that, while its configuration memory is being reprogrammed, the PLD typically cannot preserve data stored in its volatile memory (e.g., volatile embedded random access memory (RAM) blocks) and, consequently, the data is lost during the reprogramming process. However, depending upon the particular application, a user of the PLD may prefer to preserve the data stored in the volatile memory for use within the PLD after the reprogramming (i.e., reconfiguration) of the PLD has been completed and the PLD is operating based upon the new configuration data provided during the reprogramming. As a result, there is a need for improved reconfiguration techniques for PLDs.

SUMMARY

In accordance with one embodiment of the present invention, a programmable logic device includes a plurality of logic blocks; a plurality of input/output blocks; a volatile memory adapted to store data within the programmable logic device; configuration memory adapted to store first configuration data for configuration of the logic blocks, the input/output blocks, and the volatile memory of the programmable logic device; and a non-volatile memory adapted to store data provided from the volatile memory.

In accordance with another embodiment of the present invention, a programmable logic device includes a plurality of logic blocks; a volatile memory block adapted to store data within the programmable logic device; configuration memory adapted to store first configuration data for configuration of the logic blocks and the volatile memory block of the programmable logic device; means for storing in a non-volatile manner data from the volatile memory block; and means for controlling transfer of data from the volatile memory block to the storing means and from the storing means to the volatile memory block.

In accordance with another embodiment of the present invention, a method of storing information within a programmable logic device includes providing a volatile memory block; providing configuration memory adapted to configure the programmable logic device based on data stored in the configuration memory; providing a non-volatile memory block partitioned into a configuration portion and a user portion, wherein the configuration portion is adapted to store configuration data for transfer to the configuration memory to configure the programmable logic device, and wherein the user portion is adapted to store data for the volatile memory block; and controlling transfer of configuration data from the non-volatile memory block to the configuration memory and transfer of data from the volatile memory block to the non-volatile memory block and from the non-volatile memory block to the volatile memory block.

The scope of the invention is defined by the claims, which are incorporated into this section by reference. A more complete understanding of embodiments of the present invention will be afforded to those skilled in the art, as well as a realization of additional advantages thereof, by a consideration of the following detailed description of one or more embodiments. Reference will be made to the appended sheets of drawings that will first be described briefly.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** shows a block diagram illustrating an exemplary programmable logic device in accordance with an embodiment of the present invention.

FIG. **2** shows a block diagram illustrating exemplary implementation details for a non-volatile memory and configuration memory of the programmable logic device of FIG. **1** in accordance with an embodiment of the present invention.

FIG. **3** shows a block diagram illustrating exemplary implementation details for a non-volatile memory and a volatile memory of the programmable logic device of FIG. **1** in accordance with an embodiment of the present invention.

FIGS. **4** and **5** show block diagrams illustrating exemplary implementation details for a programmable logic device in accordance with an embodiment of the present invention.

FIG. **6** shows a block diagram illustrating exemplary implementation details of a non-volatile memory controller for the programmable logic device of FIG. **1** in accordance with an embodiment of the present invention.

FIG. **7** show a block diagram illustrating exemplary implementation details for data transfer between a non-volatile memory and volatile memory for the programmable logic device of FIG. **1** in accordance with an embodiment of the present invention.

Embodiments of the present invention and their advantages are best understood by referring to the detailed description that follows. It should be appreciated that like reference numerals are used to identify like elements illustrated in one or more of the figures.

DETAILED DESCRIPTION

FIG. **1** shows a block diagram illustrating an exemplary programmable logic device (PLD) **100** in accordance with an embodiment of the present invention. PLD **100** includes input/output (I/O) blocks **102** and programmable logic blocks **104**. I/O blocks **102** provide I/O functionality (e.g., supports one or more I/O and/or memory interface standards) for PLD **100**. Programmable logic blocks **104** (e.g., also referred to in the art as configurable logic blocks or logic array blocks) provide logic functionality for PLD **100**, such as LUT-based logic typically associated with FPGAs or array-based logic typically associated with CPLDs.

PLD **100** may also include non-volatile memory **106** (e.g., blocks of EEPROM or flash memory), volatile memory **108** (e.g., block SRAM), clock-related circuitry **110** (e.g., PLL circuits), one or more data ports **112**, configuration memory **114**, and/or an interconnect **116**. It should be understood that the number and placement of the various elements, such as I/O blocks **102**, logic blocks **104**, non-volatile memory **106**, volatile memory **108**, clock-related circuitry **110**, data port **112**, configuration memory **114**, and interconnect **116**, is not limiting and may depend upon the desired application. Furthermore, it should be understood that the elements are illustrated in block form for clarity and that certain elements, such as configuration memory **114** and interconnect **116**, would typically be distributed throughout PLD **100**, such as in and between logic blocks **104**, to perform their conventional functions (e.g., storing configuration data that configures PLD **100** and providing routing resources, respectively).

Data port **112** may be used for programming PLD **100**, as would be understood by one skilled in the art. For example, data port **112(1)** may represent a programming port such as a central processing unit (CPU) port, also referred to as a peripheral data port or a sysCONFIG programming port. Data port **112(2)** may represent, for example, a programming port such as a joint test action group (JTAG) port by employing standards such as Institute of Electrical and Electronics Engineers (IEEE) 1149.1 or 1532 standards. Data ports **112(1)** and **112(2)** are not both required, but one or the other or both may be included to receive configuration data and commands.

Non-volatile memory **106** may be used to store configuration data within PLD **100** for transfer to configuration memory **114** of PLD **100** upon power up or during reconfiguration of PLD **100**. This may drastically reduce the time to reconfigure PLD **100** relative to an external bitstream (e.g., reduce the time from seconds to microseconds for loading of configuration data into configuration memory **114**).

Non-volatile memory **106** may also be used to provide background programming and/or storage for PLD **100** in accordance with some embodiments of the present invention. For example for storage functionality, non-volatile memory **106** may be used to store data from volatile memory **108** during the reconfiguration process. Thus as an example, the data stored by volatile memory **108** within PLD **100**, just prior to reconfiguration, may be preserved in non-volatile memory **106** during the reconfiguration process and then written back into volatile memory **108** prior to returning to user mode, as explained further herein.

For example for background programming, PLD **100** may remain in user mode, based on the configuration data stored in configuration memory **114** within PLD **100**, while non-volatile memory **106** is programmed with new configuration data (e.g., a new user defined pattern). Once the new configuration data is stored in non-volatile memory **106**, this data can be transferred from non-volatile memory **106** to configuration memory **114** to reconfigure PLD **100**, a process sometimes referred to as refresh. The refresh process can be initiated by a signal or instruction provided to data port **112** (e.g., pulsing data port **112(1)** or providing a JTAG refresh instruction via data port **112(2)**).

As a specific example, FIG. **2** shows a block diagram illustrating a PLD **200**, which provides exemplary implementation details for PLD **100** of FIG. **1** in accordance with an embodiment of the present invention. PLD **200** includes non-volatile memory **106** (e.g., flash memory), configuration memory **114**, and control logic **202**.

Configuration memory **114** (e.g., volatile SRAM cells or other types of volatile or non-volatile memory) are used in a conventional manner to store configuration data, which determines the user defined functions of PLD **200** (e.g., determines programmable functions of I/O blocks **102**, logic blocks **104**, and interconnect **116**). Control logic **202** controls the internal transfer of the configuration data from non-volatile memory **106** to configuration memory **114**, as would be understood by one skilled in the art.

It should be understood that flash memory represents an exemplary type of memory for non-volatile memory **106**, but other types of non-volatile memory (e.g., EECMOS) that can be reprogrammed once or repeatedly may be substituted for non-volatile memory **106**. Furthermore, either non-volatile memory **106** or configuration memory **114** may be programmed (i.e., receive and store information in its memory) to store configuration data for PLD **200**, but the device functionality of PLD **200** is determined by the information stored in configuration memory **114**. Thus, PLD **200** may be configured (including reconfiguration or partial reconfiguration), for example, when information is programmed into configuration memory **114**.

It should also be understood, in accordance with one or more embodiments of the present invention, that non-volatile memory **106** and configuration memory **114** may each be programmed (including reprogrammed), for example, via data port **112(1)** or data port **112(2)**, depending upon the desired application or design requirements. Further details regarding programming may be found in U.S. Pat. No. 6,828,823 and U.S. Patent Publication No. 2005-0189962-A1, published Sep. 1, 2005.

In general, during programming of configuration memory **114** and reconfiguration of the PLD (e.g., PLD **100**), it may be desired by a user of the PLD to preserve data stored in volatile memory **108**. However for conventional approaches, any information stored in volatile memory **108** generally cannot be saved and is lost during the reconfiguration process. Furthermore, if flash memory is embedded within the PLD to provide non-volatile memory, the flash memory typically is limited. For example, the flash memory may have an inadequate, fixed amount of storage space, a slow read/write access speed, a limited number of possible read/write cycles, and/or undesirable data structure restrictions.

In contrast, in accordance with one or more embodiments of the present invention, techniques are disclosed to preserve the data stored in volatile memory **108** during a reconfiguration. For example, non-volatile memory **106** may be partitioned to provide storage for configuration data (e.g., a configuration flash section) and to provide non-volatile storage (i.e., a user flash section) for volatile memory **108** (e.g., to preserve the data through a reconfiguration, if desired by a user of the PLD). Furthermore in accordance with some embodiments it should be understood that, rather than partitioning one non-volatile memory **106**, one or more of non-volatile memory **106** may be implemented to provide non-volatile storage for volatile memory **108**, while one or more of non-volatile memory **106** may be implemented to provide non-volatile storage for configuration data.

Thus, for example for some embodiments, non-volatile memory **106** may be used as non-volatile storage for a user to store not only various information, such as system management information, manufacturing control information, and/or failure statistics information for board level diagnostics, but also to store in a non-volatile fashion data from volatile memory **108** to preserve during a reconfiguration or power down of PLD **100** (also referred to as store-to-flash capability). Furthermore, in accordance with one or more embodiments of the present invention, one or more blocks of volatile memory **108** may be operated as shadow flash memory during user mode of operation to provide fast, random, and unlimited read/write access as well as a flexible, scalable data structure capability. Consequently, in accordance with some embodiments, volatile memory **108** may function as virtual user flash memory with store-to-flash capability to preserve the data during a reconfiguration (e.g., transparent field reconfiguration capability).

For example, FIG. **3** shows a PLD **300**, which illustrates exemplary implementation details for a non-volatile memory **302** and volatile memory **108** (e.g., labeled EBR for embedded block RAM) of PLD **100** of FIG. **1** in accordance with an embodiment of the present invention. PLD **300** illustrates the general functionality for non-volatile memory **302** and volatile memory **108** in accordance with some embodiments. For example, non-volatile memory **302** may be used to store data from volatile memory **108**, which allows volatile memory **108** to be used as virtual flash memory (e.g., also referred to as shadow flash) during a user mode of operation.

Specifically as an example, non-volatile memory **302** may represent a specific implementation example for non-volatile memory **106**, with non-volatile memory **302** partitioned (segmented) into a user flash portion **304**, a trim portion **306**, a test portion **308**, and a configuration flash portion **310** (with trim portion **306** and test portion **308** being optional portions within non-volatile memory **302**). User flash portion **304** may be used to store data for volatile memory **108** in a non-volatile manner, such as to preserve the data stored by volatile memory **108** during a reconfiguration of PLD **300** (e.g., a store-to-flash (STF) operation).

Trim portion **306** may be used to store trim data, which may be used for adjusting or trimming various parameters within PLD **300** (e.g., adjusting current source values, resistor values, or other circuit parameters as would be understood by one skilled in the art). Test portion **308** (e.g., also referred to as program electronic signature (PES) in one embodiment) may be used to store test data and other types of information, as desired. Configuration flash portion **310** may be used to store configuration data for transfer to configuration memory **114** to reconfigure PLD **300**.

As shown in FIG. **3**, non-volatile memory **302** (e.g., flash memory) may have a different starting address for each portion or section. User flash portion **304** may be accessed sequentially and coupled (e.g., via routing resources **312**, such as with multiplexers and other circuitry associated with interconnect **116**) to its associated volatile memory **108**. Therefore, data can be loaded into volatile memory **108** from user flash portion **304** (e.g., during reconfiguration or as desired) and volatile memory **108** can be accessed randomly during user mode of operation. If all or a portion of PLD **300** which includes volatile memory **108** is to be reconfigured, the data in volatile memory **108** which is desired to be preserved through the reconfiguration process may be loaded into user flash portion **304** (e.g., under control of a store-to-flash (STF) command).

In general in accordance with an embodiment of the present invention, non-volatile memory **302** illustrates the partitioning of flash memory into configuration flash portion **310** (e.g., a configuration flash area) and user flash portion **304** (e.g., a user flash area), which allows volatile memory **108** to function, if desired by a user, as shadow flash memory (e.g., virtual flash memory) with STF capability. Configuration flash portion **310** may be used to store and provide configuration data to configuration memory **114** (e.g., provide initializing configuration logic), while user flash portion **304** may be used to store data for volatile memory **108** and provide data for initializing volatile memory **108** (e.g., volatile memory **108** is the primary interface with user flash portion **304**). Non-volatile memory **302** may further include additional areas (e.g., trim portion **306** and test portion **308**) to provide additional memory for users to store other information based on their specific requirements or application.

The techniques illustrated herein may provide certain advantages over a conventional PLD. For example, by utilizing volatile memory **108** as virtual flash memory, there may be certain benefits over conventional techniques in terms of power consumed, speed (e.g., fast accessibility), accessibility (e.g., random, unlimited read/write), and data structure (e.g., flexible data structure and organization). By utilizing volatile memory **108** as shadow user flash, less power may be consumed relative to using non-volatile memory **302** directly. Volatile memory **108** can function as shadow user flash and provide virtual user flash capability as desired by a user due to the STF capability for user flash portion **304**. The amount of shadow user flash provided by PLD **300** may vary and scale with device density (e.g., from 180 k bits to 400 k bits or more) and may be organized as one or more separate shadow user flash blocks, as described further herein.

Volatile memory **108** may also provide speed advantages relative to flash technology. For example, shadow user flash may be accessed at the speed of the volatile memory technology of volatile memory **108** rather than the relatively slow read/write speeds of non-volatile memory **302**. Thus, by using volatile memory **108** as shadow user flash, virtual non-volatile storage space may be provided at the speed of SRAM technology, for example, if volatile memory **108** is implemented as SRAM.

Volatile memory **108** may also provide accessibility advantages over non-volatile memory **302**. For example, volatile memory **108** may provide random read/write access capability as shadow user flash, rather than the typically more restricted sequential access of non-volatile memory **302** (e.g., implemented as flash technology). Furthermore for example, volatile memory **108** may provide essentially unlimited read/write access capability, rather than the typically more restrictive endurance limitations applied to the read/write access capability of non-volatile memory **302** (e.g., not limited by the endurance limitations of flash technology).

Volatile memory **108** may also provide flexible data structure, organization, and access capability. For example, by using volatile memory **108** (e.g., SRAM block), the conventional routing and addressing structure may be utilized to provide flexible data access, with volatile memory **108** providing flexible data width organization and, for multiple blocks of volatile memory **108**, cascading for both width and depth data structure flexibility. For example, volatile memory **108** may provide various data structure access flexibility (e.g., by 1, by 2, by 4, by 8, by 16, by 32, by 36, etc., such as 16K by 1, 8K by 2, 4K by 4, 2K by 8, 1K by 18, 512 by 36, etc.).

The techniques disclosed herein may provide additional benefits or advantages. For example, there does not have to be any special-purpose or dedicated routing and interconnect to use volatile memory **108** as shadow user flash, but rather the PLD's inherent routing structure (e.g., interconnect **116**) may be used. Furthermore, the use of volatile memory **108** as shadow user flash is flexible and user-controlled by providing STF control capability for saving the information stored in volatile memory **108** to non-volatile memory **302** (e.g., flash memory).

FIGS. **4** and **5** show block diagrams of PLD **400** and PLD **500**, respectively, illustrating exemplary implementation details for a PLD (e.g., PLD **100**) in accordance with an embodiment of the present invention. For example, PLD **400** illustrates a non-volatile memory **406** (e.g., similar to non-volatile memory **106** or non-volatile memory **302**), volatile memory **108**, and logic portions **402**. Logic portions **402** represent various programmable circuitry of PLD **400** and may include, for example, I/O blocks **102**, programmable logic blocks **104**, configuration memory **114**, and interconnect **116**.

Non-volatile memory **406** may be partitioned into user flash portion **304** (UFM) and configuration flash portions **310** (labeled CFM and separately referenced as configuration flash portions **310(1)** and **310(2)**). Non-volatile memory **406** may further include a tag portion **404**, which may represent a small bank of memory for use as a user desires (e.g., a scratch-pad memory).

Configuration flash portion **310(1)** may be used to store the configuration data for transfer to configuration memory **114** within logic portions **402(1)** and **402(2)**, while configuration flash portion **310(2)** may be used to store the configuration data for transfer to configuration memory **114** within logic portions **402(3)** and **402(4)** to configure those portions of PLD **400**. User flash portion **304** may be associated with volatile memory **108(1)** and/or **108(2)**. For example, volatile memory **108(1)** and **108(2)** (e.g., embedded block RAM or EBR) may be mapped into user flash portion **304** and, as explained further herein, a STF operation may be performed (e.g., during a reconfiguration process) to save the information stored in volatile memory **108(1)** and **108(2)** to user flash portion **304**. As a specific example, the information transferred from volatile memory **108(1)** and **108(2)** to user flash portion **304** may then be transferred back from user flash portion **304** to volatile memory **108(1)** and **108(2)** during the reconfiguration process but prior to entering user mode of operation so that the information in volatile memory **108(1)** and **108(2)** is preserved during the reconfiguration process.

PLD **500** of FIG. **5** illustrates that any number of non-volatile memory **406** and volatile memory **108** may be provided and, furthermore, each volatile memory **406** may be partitioned as desired. For example, configuration flash portion **310(1)** may be used to store the configuration data for transfer to configuration memory **114** within logic portion **402(1)**, while configuration flash portion **310(5)** may be used to store the configuration data for transfer to configuration memory **114** within logic portion **402(2)**.

In a similar fashion, user flash portion **304(1)** may be associated with and mapped to volatile memory **108(1)**, while user flash portion **304(3)** may be associated with and mapped to volatile memory **108(2)**. Likewise, configuration flash portions **310(2)** and **310(3)**, **310(4)**, **310(6)** and **310(7)**, and **310(8)** may be associated with logic portions **402(3)**, **402(5)**, **402(4)**, and **402(6)**, respectively, while user flash portions **304(2)** and **304(4)** may be associated with and mapped to volatile memory **108(3)** and **108(4)**, respectively. Thus, for example, shadow user flash memory may be provided of variable density, with the amount of shadow user flash available related to the number or amount of volatile memory **108** in the PLD and which may scale with density. Furthermore in accordance with some embodiments, it should be understood that each volatile memory **108** within PLD **500** may be configured independently as shadow user flash or as any other type of conventional memory implementation (e.g., as ROM, RAM, etc.).

In accordance with an embodiment of the present invention, volatile memory **108** may function as the primary interface for user flash portion **304**. For example, during user mode of operation, direct access to user flash portion **304** may only be through the associated volatile memory **108** (shadow user flash).

FIG. **6** shows a block diagram illustrating a PLD **600**, which provides exemplary implementation details for a non-volatile memory controller **602** for PLD **100** of FIG. **1** in accordance with an embodiment of the present invention. PLD **600** illustrates, for example, the STF capability for PLD **100** as discussed herein.

For example, controller **602** controls user flash portions **304** (e.g., **304(1)** and **304(2)**) for transferring information between volatile memory **108** and user flash portions **304** and may be separate from or be part of control logic **202**. As an example, control logic **202** and controller **602** control the transfer of configuration data from non-volatile memory **106** to configuration memory **114** and also control the transfer of information between non-volatile memory **106** (e.g., user flash portions **304**) and volatile memory **108**.

As a specific implementation example, a store-to-flash signal may be provided to controller **602** to command the information stored in volatile memory **108** to be transferred to the associated mapped user flash portion **304**. The store-to-flash signal may be provided, for example, via a common interface block (CIB) **604**. As another specific implementation example, in addition to the store-to-flash signal, a flash busy signal and a status signal may be provided by controller **602** to indicate if a flash related operation is in progress (e.g., store-to-flash process) and operations status (e.g., whether the store-to-flash operation passed or failed), respectively.

Each user flash portion **304** may have an associated configuration bit **606** (e.g., one of configuration memory cells **114**) for indicating to controller **602** whether associated user flash portion **304** is to be used in the STF capacity (i.e., user flash mode). For example, configuration bits **606(1)** and **606(2)** are associated with user flash portions **304(1)** and **304(2)**, respectively. Therefore in accordance with one embodiment, controller **602** saves the information in volatile memory **108** based upon the store-to-flash signal and the values stored in configuration bits **606**.

Additional information and control signals may be used, as illustrated in FIG. **6**. For example, rather than just the store-to-flash signal, controller **602** may instead or in addition receive a flash erase signal **608** and a transfer-to-flash signal **610** and provide an erase done signal **612** and a transfer done signal **614**.

Flash erase signal **608** would be used to command controller **602** to erase user flash portions **304** whose configuration bit **606** is appropriately set to designate user flash mode for that user flash portion **304**. Once the erasure has been completed, controller **602** may indicate this (e.g., to user logic) via erase done signal **612**. Transfer-to-flash signal **610** would then be used to command controller **602** to transfer the information in volatile memory **108** to the associated user flash portion **304**. Once the transfer has been completed, controller **602** may indicate this (e.g., to user logic) via transfer done signal **614**.

The information stored in user flash portions **304** may be protected from unauthorized access in the same way as is conventionally done for protecting configuration data from unauthorized access from non-volatile memory with the PLD. For example, the data stored in user flash portions **304** and configuration flash portions **310** may be secured from unauthorized reading out by a conventional security fuse, which prevents the reading out from the PLD of the information stored in the embedded non-volatile memory.

FIG. **7** show a block diagram illustrating a PLD **700**, which provides exemplary implementation details for data transfer between a non-volatile memory **704** and volatile memory **108** for PLD **100** of FIG. **1** in accordance with an embodiment of the present invention. Non-volatile memory **704** for this exemplary implementation may represent four user flash portions **304** (i.e., user flash portions **304(1)** through **304(4)**) that correspond to volatile memory **108(1)** through **108(4)**. Control logic **702** may represent controller **602** (FIG. **6**) or represent controller **602** and control logic **202** (FIG. **1**) and it should be understood that other circuitry within PLD **700** is not shown for clarity.

Volatile memory **108(1)** through **108(4)**, for example, have been configured by a user pattern to function as user flash memory, RAM, RAM, and ROM, respectively. Consequently in accordance with an embodiment of the present invention, user flash portions **304(1)** through **304(4)** may provide non-volatile storage for corresponding volatile memory **108(1)** through **108**(**4**) if, for example, the associated configuration bit **606** designates user flash mode.

FIG. **7** illustrates, with the bold arrow from non-volatile memory **704** through volatile memory **108**, the ability to transfer data from non-volatile memory **704** to volatile memory **108** to initialize volatile memory **108** with desired data. For example, control logic **702** may provide an enable signal **712** (labeled EBR enable) to volatile memory **108** to enable volatile memory **108** to store the data provided by non-volatile memory **704**.

Control logic **702** may be controlled, for example, via SPI or JTAG ports (e.g., via data port **112**) during the store-to-flash operation or to initiate the store-to-flash operation. Alternatively for example, a store signal **708** provided to control logic **702** may include flash erase signal **608** and transfer-to-flash signal **610**, while control logic **702** may provide a store status signal **706** that includes erase done signal **612** and transfer done signal **614**, with these signals used as described in reference to FIG. **6**.

PLD **700** may be viewed as operating in a user mode of operation as configured by a user pattern (i.e., a user defined pattern “A”). If a user desires to reconfigure PLD **700**, non-volatile memory **704** may be erased (e.g., write all logical high values “1” by command of flash erase signal **608**) and then a new user pattern (i.e., a user defined pattern “B”) may be written to non-volatile memory **704**, while PLD **700** operates in user mode with user defined pattern “A” (e.g., background programming operation performed). For example, non-volatile memory **704** may now store a default pattern (e.g., store all logical low values “0”), except for section **304**(**4**) corresponding to volatile memory block **108**(**4**) that a user desires to function as ROM and thus section **304**(**4**) stores the desired ROM data (e.g., the same ROM data or different ROM data as in user defined pattern “A”).

Prior to transferring the user defined pattern “B” into configuration memory **114** from configuration flash portions **310**, a user may desire to preserve the data stored in volatile memory **108**. Therefore, as illustrated in FIG. **7** by the bold arrow from volatile memory **108** to non-volatile memory **704**, a store-to-flash operation (e.g., by command of transfer-to-flash signal **610**) may be performed to transfer the data stored in volatile memory **108**(**1**), **108**(**2**), **108**(**3**), and/or **108**(**4**) to user flash portions **304**(**1**), **304**(**2**), **304**(**3**), and **304**(**4**), respectively, to preserve the data during the reconfiguration process.

During the store-to-flash operation, control logic **702** may provide a store control signal **714** to volatile memory **108** and a flash enable control signal **710** to non-volatile memory **704** to control the transfer and storage of data from volatile memory **108** to non-volatile memory **704**. Upon completion of the store-to-flash operation, the refresh operation may be performed to reconfigure PLD **700**. For example for the refresh operation, the configuration data stored in configuration flash section **310** may be transferred to configuration memory **114**, and the data stored and preserved in non-volatile memory **704** may be transferred to volatile memory **108**. The refresh operation may be initiated, for example, by powering down and then powering up PLD **700**, toggling a program pin of PLD **700**, or sending a refresh command to PLD **700** (e.g., via a slave SPI interface or JTAG port).

Systems and methods are disclosed herein to provide improved non-volatile storage techniques for programmable logic devices. For example, in accordance with an embodiment of the present invention, a non-volatile PLD (e.g., FPGA) with user flash is disclosed that provides for the preservation of user data within volatile memory blocks. For example in accordance with some embodiments, the PLD includes flash memory that is partitioned into configuration flash and user flash sections, with the user flash sections associated with the volatile memory blocks. Consequently, the volatile memory blocks may be configured to function as shadow user flash, with the information transferred to the corresponding user flash sections when desired by a user to preserve the information (e.g., during a reconfiguration or power down of the PLD).

Embodiments described above illustrate but do not limit the invention. It should also be understood that numerous modifications and variations are possible in accordance with the principles of the present invention. Accordingly, the scope of the invention is defined only by the following claims.

What is claimed is:

1. A programmable logic device comprising:

a plurality of logic blocks;

a plurality of input/output blocks;

a volatile memory block adapted to store user data within the programmable logic device, but which is not adapted to configure the logic blocks and the input/output blocks;

configuration memory adapted to store first configuration data for configuration of the logic blocks, the input/output blocks, and the volatile memory block of the programmable logic device; and

a non-volatile memory adapted to store the user data provided from the volatile memory block and further adapted to store configuration data for transfer to the configuration memory, the non-volatile memory being partitioned to provide a configuration portion to store configuration data for transfer to the configuration memory and a user portion to store the user data for the volatile memory block,

wherein the volatile memory block and the configuration memory are distinct and separate from each other and located in different portions of the programmable logic device.

2. The programmable logic device of claim **1**, wherein the volatile memory block includes embedded block RAM.

3. The programmable logic device of claim **1**, further comprising control logic adapted to control transfer of the user data from the volatile memory block to the non-volatile memory and from the non-volatile memory to the volatile memory block.

4. The programmable logic device of claim **1**, further comprising a non-volatile memory controller adapted to control transfer of the user data from the volatile memory block to the non-volatile memory and from the non-volatile memory to the volatile memory block.

5. The programmable logic device of claim **1**, wherein the non-volatile memory is further partitioned to provide at least one of a trim portion, a test portion, and a tag portion.

6. The programmable logic device of claim **1**, wherein the configuration portion and the user portion each has a different starting address.

7. The programmable logic device of claim **1**, further comprising at least a first data port adapted to receive commands for controlling reconfiguration of the programmable logic device.

8. The programmable logic device of claim **1**, wherein the volatile memory block is further adapted to function as virtual flash memory, wherein at least one of the configuration memory is associated with the volatile memory block and provides a control signal to control whether the volatile memory block transfers its user data to the non-volatile memory.

9. The programmable logic device of claim **1**, wherein the non-volatile memory is flash memory partitioned into a configuration flash portion and a user flash portion.

10. A programmable logic device comprising:

a plurality of logic blocks;

a plurality of input/output blocks;

a volatile memory block adapted to store user data within the programmable logic device, but which is not adapted to configure the logic blocks and the input/output blocks;

configuration memory adapted to store first configuration data for configuration of the logic blocks, the input/output blocks, and the volatile memory block of the programmable logic device; and

non-volatile memory partitioned to provide a configuration portion to store configuration data for transfer to the configuration memory and a user portion to store the user data for the volatile memory block,

wherein the volatile memory block and the configuration memory are distinct and separate from each other and located in different portions of the programmable logic device, and the volatile memory block is further adapted to function as virtual flash memory, with the non-volatile memory accessed via the volatile memory block.

11. A programmable logic device comprising:

a plurality of logic blocks;

a volatile memory block adapted to store user data within the programmable logic device, wherein the volatile memory block is not adapted to configure the logic blocks;

configuration memory adapted to store first configuration data for configuration of the logic blocks and the volatile memory block of the programmable logic device;

means for storing in a non-volatile manner the user data from the volatile memory block and configuration data for the configuration memory, wherein the storing means is partitioned to provide a configuration portion to store configuration data for transfer to the configuration memory and a user portion to store the user data for the volatile memory block; and

means for controlling transfer of the user data from the volatile memory block to the storing means and from the storing means to the volatile memory block and for controlling transfer of configuration data from the storing means to the configuration memory,

wherein the volatile memory block and the configuration memory are distinct and separate from each other and located in different portions of the programmable logic device.

12. The programmable logic device of claim **11**, further comprising means for communicating with the controlling means.

13. A programmable logic device comprising:

a plurality of logic blocks;

a volatile memory block adapted to store user data within the programmable logic device, wherein the volatile memory block is not adapted to configure the logic blocks;

configuration memory adapted to store first configuration data for configuration of the logic blocks and the volatile memory block of the programmable logic device;

means for storing in a non-volatile manner the user data from the volatile memory block and configuration data for the configuration memory, wherein the storing means is partitioned to provide a configuration portion to store configuration data for transfer to the configuration memory and a user portion to store the user data for the volatile memory block; and

means for controlling transfer of the user data from the volatile memory block to the storing means and from the storing means to the volatile memory block,

wherein the volatile memory block and the configuration memory are distinct and separate from each other and located in different portions of the programmable logic device; and

wherein at least one cell of the configuration memory is associated with the volatile memory block and provides a control signal to control whether the volatile memory block can transfer its user data to the storing means.

14. A method of storing information within a programmable logic device, the method comprising:

providing a volatile memory block adapted to store user data, wherein the volatile memory block is not adapted to configure the programmable logic device;

providing configuration memory adapted to configure the programmable logic device based on data stored in the configuration memory;

providing a non-volatile memory block partitioned into a configuration portion and a user portion, wherein the configuration portion is adapted to store configuration data for transfer to the configuration memory to configure the programmable logic device, and wherein the user portion is adapted to store user data for the volatile memory block; and

controlling transfer of configuration data from the non-volatile memory block to the configuration memory and transfer of the user data from the volatile memory block to the non-volatile memory block and from the non-volatile memory block to the volatile memory blocks,

wherein the volatile memory block and the configuration memory are distinct and separate from each other and located in different portions of the programmable logic device.

15. The method of claim **14**, wherein the volatile memory block is configurable to function as shadow user non-volatile memory and is associated with the user portion of the non-volatile memory block.

16. The method of claim **14**, further comprising a plurality of the volatile memory blocks, wherein the non-volatile memory block is further partitioned into a plurality of user portions associated with the volatile memory blocks, and wherein the non-volatile memory block is further partitioned into at least one of a trim portion, a test portion, and a tag portion.

17. The method of claim **14**, further comprising performing a background programming operation prior to the transfer of the user data from the volatile memory block to the non-volatile memory block to preserve the user data from the volatile memory block during a reconfiguration.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 7,554,358 B1
APPLICATION NO. : 11/397985
DATED : June 30, 2009
INVENTOR(S) : Fabiano Fontana

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Col. 12, claim 14, line 20: change "the non-volatile memory block to the volatile memory blocks," to --the non-volatile memory block to the volatile memory block,--

Signed and Sealed this

Twenty-seventh Day of April, 2010

David J. Kappos

David J. Kappos
Director of the United States Patent and Trademark Office