Design and implementation of a programming circuit in radiation-hardened FPGA

Wu Lihua(吴利华)†, Han Xiaowei(韩小炜), Zhao Yan(赵岩), Liu Zhongli(刘忠立), Yu Fang(于芳), and Stanley L. Chen(陈陵都)

Institute of Semiconductors, Chinese Academy of Sciences, Beijing 100083, China

Abstract: We present a novel programming circuit used in our radiation-hardened field programmable gate array (FPGA) chip. This circuit provides the ability to write user-defined configuration data into an FPGA and then read it back. The proposed circuit adopts the direct-access programming point scheme instead of the typical long token shift register chain. It not only saves area but also provides more flexible configuration operations. By configuring the proposed partial configuration control register, our smallest configuration section can be conveniently configured as a single data and a flexible partial configuration can be easily implemented. The hierarchical simulation scheme, optimization of the critical path and the elaborate layout plan make this circuit work well. Also, the radiation hardened by design programming point is introduced. This circuit has been implemented in a static random access memory (SRAM)-based FPGA fabricated by a 0.5 μm partial-depletion silicon-on-insulator CMOS process. The function test results of the fabricated chip indicate that this programming circuit successfully realizes the desired functions in the configuration and read-back. Moreover, the radiation test results indicate that the programming circuit has total dose tolerance of 1×10^5 rad(Si), dose rate survivability of 1.5×10^{11} rad(Si)/s and neutron fluence immunity of 1×10^{14} n/cm^2.

Key words: FPGA; configuration; read back; partial configuration; partial-depletion SOI; radiation-hardened
DOI: 10.1088/1674-4926/32/8/085012 **EEACC:** 1265A

1. Introduction

As process technology and device architecture develop, the logic density and system performance of SRAM-based FPGAs continue to increase[1, 2]. In addition, the most charming reconfigurable feature of FPGAs provides the ability to implement system-level prototype design and specific application with lower non recurring engineering (NRE) costs and fast time-to-market. All of these characteristics make SRAM-based FPGAs widely adapted in communications, network and storage systems applications. However, most commercial SRAM-based FPGAs fabricated by a bulk-CMOS process cannot meet the requirements of aerospace, military and avionic applications because they are more susceptible to single event upset and transient radiation effects. Silicon-on-insulator (SOI) technology has been introduced to FPGAs[3−5] considering its better radiation hardened performance, better latch-up immunity and smaller parasitic capacitance[6, 7].

A programming circuit in an FPGA is essentially used to download the user-defined bitstream into the FPGA[7, 8]. The proposed novel programming (PGM) circuit in this paper is designed for our radiation-hardened FPGA fabricated PD SOI CMOS process. In our design, this circuit provides a standard and prevalent joint test access group (JTAG) configuration interface, conveniently fulfilling prototype design. Also, it is very compatible with Xilinx Platform configuration flash PROMs. Compared with current-commercial FPGA chips, the smallest configuration section of which is one data frame, this circuit could write each programming point (PP) individually in an FPGA by configuring the smallest configuration section as a single data so as to provide more flexible configuration operations, which can efficiently resolve the memory coherence problem in FPGAs[8]. Furthermore, the radiation hardened consideration for programming point by design is introduced to improve the radiation hardened performance. Also, a hierarchical simulation scheme, the optimization of the critical path and the elaborate layout plan make this circuit work well.

2. Programming circuit overview

As illustrated in Fig. 1, our FPGA is composed of a logic plane and a PGM circuit. The logic plane mainly consists of a logic block (LB), an I/O cell block (IOB), a global signals (GB) block and a routing channel block containing a horizontal direction block (CBX), a vertical direction block (CBY) and a switch block (SB). The PGM circuit consists of PGM core logic and millions of PPs. After the bitstream is generated according to the application, the presented PGM circuit will load each bit into the corresponding PP to achieve user-defined logic function.

The detail functional block diagram of our PGM circuit is shown in Fig. 2. The PGM core logic is composed of an interface, a pgmctl, a dispatch and write/read module, which receives, writes and reads back configuration data, and the PPs, which store the configuration data.

The interface module provides the interface between the FPGA and the external programming hardware. The pgmctl module orchestrates the sequence of operations for configuration and read-back. The dispatch module parses the data received from the interface module and dispatches them to the corresponding configuration register. The write/read module finally writes configuration data into PPs or reads back content

† Corresponding author. Email: lihua@semi.ac.cn

Received 24 February 2011, revised manuscript received 10 March 2011

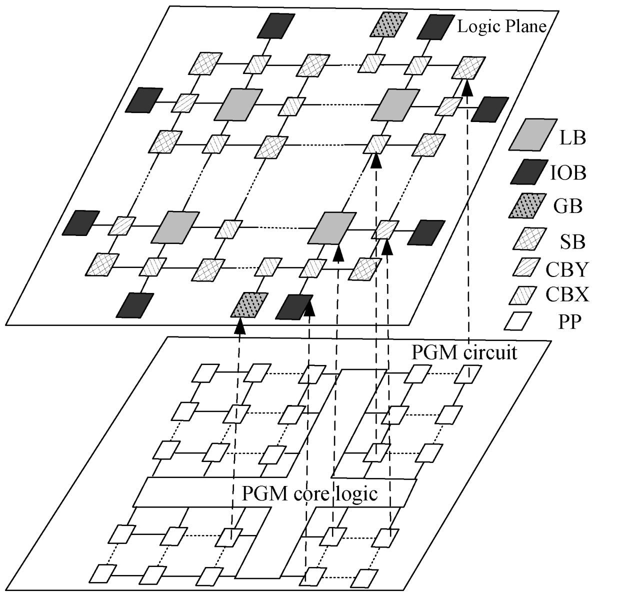


Fig. 1. Block diagram of a FPGA with a logic plane and a PGM circuit.

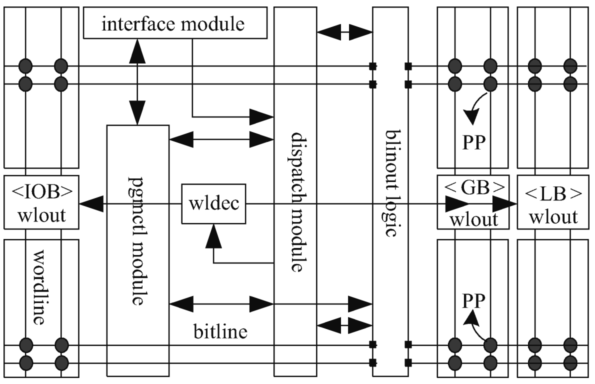


Fig. 2. Function block diagram of a PGM circuit.

from the PPs, which is divided into column decode logic and blinout logic. The column decode logic contains weldec and several wlout.

The programming circuit is designed for the following functions: (1) configuration; (2) read back; and (3) partial configuration.

3. Programming circuit design

3.1. Programming point design

The programming point in our FPGA is a five-transistor (5T) memory cell, as described in Ref. [9]. To configure the FPGA, the PPs are constructed as an array of blocks arranged by the orthogonal x (column) and y (row) coordinates. Each LB or IOB or GB occupies a single block location. The PPs are distributed to this block array structure. These PPs are treated like memory cells positioned at the cross-points of worldlines and bitlines, as depicted in Fig. 2. The PGM core circuit is responsible for writing each configuration bit to the correct PP and reading the content from the PPs. In particular, these bits located in the same column coordinate PPs are grouped as a frame data in the bitsream.

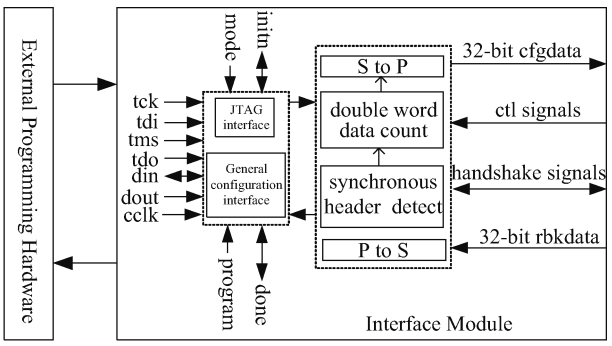


Fig. 3. Block diagram of the interface module.

3.2. Interface module design

The interface module illustrated in Fig. 3 provides the interface between the FPGA and the external programming hardware in three configuration modes: A. master-serial mode, B. slave-serial mode and E. JTAG mode.

After FPGA power-on and finishing initialization, the handshake initn pin goes high and the mode pin is sampled to determine which kind of configuration mode is selected. However, the JTAG configuration mode is prior to the other two modes. This means that JTGA mode will be chosen when JTGA configuration instruction is asserted whatever the mode pin is. The configuration data stored in the external programming hardware will be serially transferred into the interface module though a din pin or a tdi pin. These data will be packed into 32-bit data parallel when a synchronous header is detected. The 32-bit data called cfgdata is then sent to the dispatch module to execute the configuration process. After finishing configuration, the handshake done pin goes high and data transmission will halt.

Similarly, a configuration file needs to be transferred into the PGM to enable the read-back function if one executes a read-back process. Then the 32-bit data, called rbkdata, are applied to the interface module. These data will be parsed into a single data to be transferred out of din or tdo pin serially.

3.3. Pgmctl module design

The pgmctl module is a top control module of the entire programming circuit. It accomplishes its function via a central programming control state machine, as shown in Fig. 4.

The FPGA goes to pgmrst state when the FPGA power on or program pin activates. In this state, all of the configuration register will be initiated and some global signals, such as gts, gwe and gsr, should be set/reset[10, 11]. If the program pin goes high, the FPGA will go to the clrmemst state to initiate the PPs until spreading all of the location of PPs array. Then the clrdonest state is the current state and the initn pin goes high. If a configuration command wcfg_cmd_o parsed by the dispatch module is received, the FPGA will go to the pgminst state and the configuration data will begin to be configured into PPs until the start_cmd_o command is received. During the pgmdonest state, the FPGA will track the eos (end of start-up sequence[12, 13]) signal to go into the pgmidle state. Then the configuration process is finished. Also, if we want to reconfigure the FPGA online, the configuration process can be started from the pgmidle

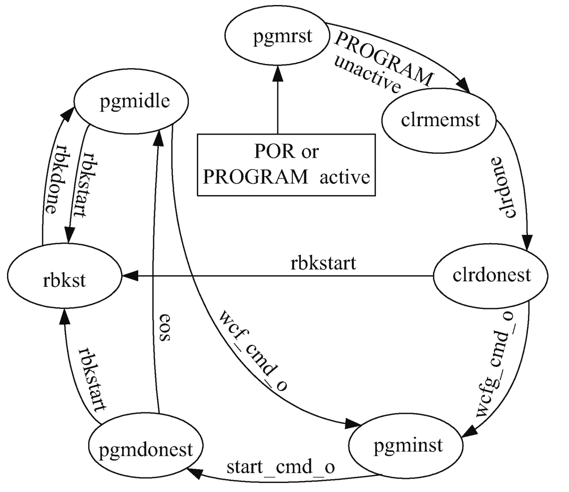


Fig. 4. The pgmctl module state diagram.

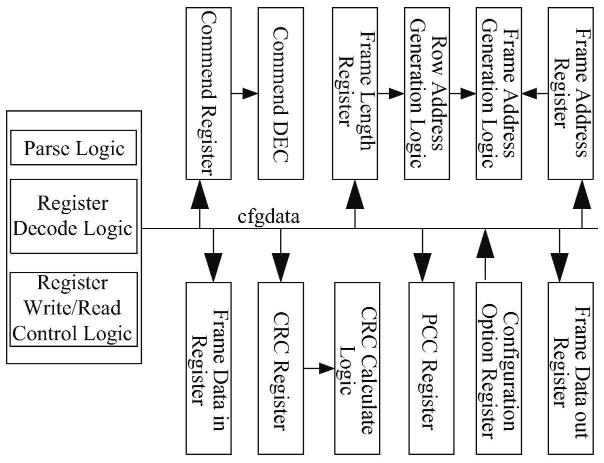


Fig. 5. Block diagram of the dispatch module.

state by the wcfg_cmd_o commend. If a rbkstart commend is received during the clrdonest, pgmdonest or pgmidle state, the FPGA will go to the rbkst state to read back the contents of the PPs until the rbkdone signal is effected.

3.4. Dispatch module design

The block diagram of the dispatch module is shown in Fig. 5. The 32-bit cfgdata assembled by the interface module is transferred to the dispatch module to be parsed into thecpacket header or packet data. Each 32-bit packet header contains some information (e.g., writing/reading option, the number of 32-bit packet data following this packet header and the destination of these packet data). Each packet data will be sent to an appropriate register, as indicated in their packet headers.

Concretely, the command register[12, 13] (CMDR) receives the current PGM commend, as indicated by the command DEC logic; the frame length register[12, 13] (FLR) receives the information about the frame, including the row location of the first 32-bit data in this frame and the length of the frame indicated by the number of 32-bit data; the row address generation logic gives the final row address of each 32-bit data in a frame by sequence; the frame address register[12, 13] (FAR) contains the column information about one frame; the frame address generation logic gives the final column address of the frame or automatically generates the other frame column addresses frame by frame; the frame data in the register[12, 13] (FDIR) stores the information downloaded into the PPs; the cyclic redundancy check register[12, 13] (CRCR) stores the original CRC value and the CRC calculate logic gives the final CRC result; the configuration option register[12, 13] (COR) determines some detailed information (e.g., FPGA start-up sequence, daisy-chain configuration selection and whether executing CRC calculation); the Partial Configuration Control Register (PCCR) is proposed to execute precise partial configuration according to the application; and the frame data out register[12, 13] (FDOR) receives the read-back data from the write/read module during the FPGA rbkst state.

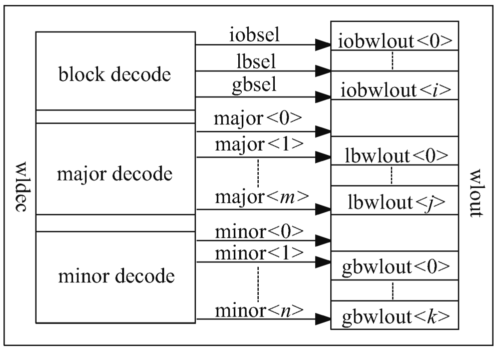


Fig. 6. Block diagram of the column decode module.

3.5. Write/read module design

3.5.1. Column decode logic

The column decode module decodes the frame address obtained from the dispatch module to an activated wordline (wl). A direct-access scheme is implemented instead of the typical long token shift register chain design[12, 14, 15]. As briefly illustrated by Fig. 6, the function of the wldec logic is to hierarchically decode the address information into block selection, major selection and minor selection, and the wlout logic receives this hierarchical decoding information to generate wordline. This scheme not only saves the area by abandoning the long shift register chain but also is particularly efficient for partial programming applications where the starting address can be any frame.

3.5.2. Blinout logic

Similar to the column decode logic, in blinout logic, we propose a direct-access scheme instead of the typical long token shift register chain design[14, 15] to achieve configuration writing and reading. As shown in Fig. 7, the blinout logic is composed of some identical row logics. We have 11 row logics considering our FPGA's one full length frame. Each row logic has its owner row location and corresponds to one 32-bit packet data within a frame. For example, row $<i>$ logic occupys the i location in the vertical direction and takes responsibility for writing and reading the 32-bit packet data in the i row location parsed from the dispatch module frame by frame. Concretely, the logics of each row are composed of 32 wr cells, as illustrated in Fig. 7, which implement the 32-bit data write/read

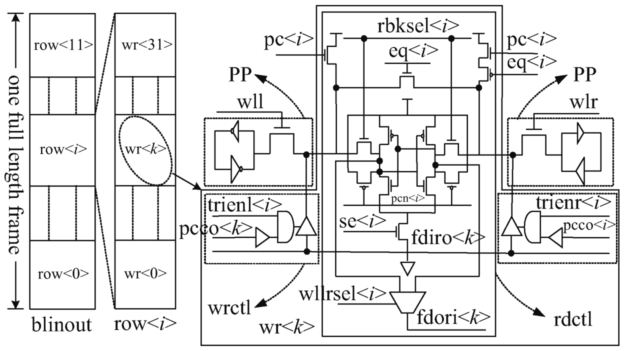


Fig. 7. Block diagram of the blinout logic.

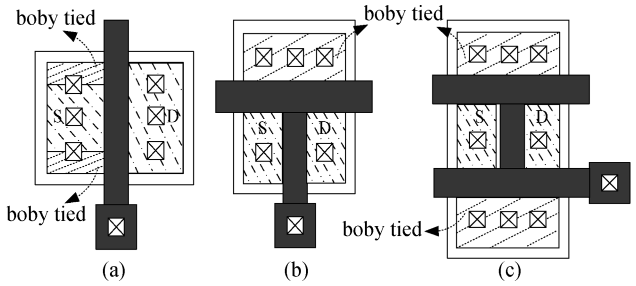


Fig. 8. (a) BTS-type body-tied structure. (b) T-type body-tied structure. (c) H-type body-tied structure.

operation one time.

3.5.3. Implementation of partial reconfiguration

In the FPGA application, particularly in the space environment, the typical single event upsets (SEUs) usually cause transient faults in the FPGA by upsetting the content of the on-chip memory cells, which define the logic functionality. The occurrence rate of SEUs for applications in the projected low earth orbital path is expected to be around one per hour across three Xilinx XCV1000 FPGA devices and is crucial for space applications[16]. The online partial reconfiguration feature makes it possible to read back and modify the error data in PPs to recover without interrupting the FPGA operation[8]. Unfortunately, if these error configuration data are grouped in the same frame with those PPs configured as read access memory (RAM) functionality whose values will vary dynamically with the process of designer applications, the simple online partial configuration will cause a memory coherence problem in a Xilinx Virtex series FPGA[8] because their smallest configuration section is one frame[12].

Unlike Xilinx Virtex series FPGAs, our PGM circuit provides the ability to configure any PP separately. Concretely, the column decode logic can identify the data frame at a horizontal position and the row address generation logic can give the row location of every 32-bit data within one frame in the vertical direction. Using this decode scheme, every 32-bit packet data can be located by the orthogonal x (column) and y (row) coordinates. Accompanying the PCCR, if we want to shield any single data within a 32-bit packet data, we can configure the corresponding data of PCCR as logic “0”. As shown in Fig. 7, if the value of pcco <k> from PCCR is logic “0”, the fdiro <k> obtained from FDIR will be shielded.

Unlike the proposed scheme[8], increasing the design overhead, using our partial reconfiguration function, the user could conveniently recover any error configuration data without stalling the FPGA operation to avoid a memory coherence problem.

3.6. Radiation hardened design

Based on the natural radiation-hardened performance of SOI material, some radiation hardened designs (RHBD) are proposed on the single event effect (SEE) sensitivity parts of a PGM circuit. As described in Ref. [17], the PPs module should be treated specially in the PGM circuit.

Considering the area overhead of PPs in the FPGA, some redundancy memory cell structure[18] is not adopted in our design while SOI body-tied techniques and an H-type gate transistor structure[19] are introduced to improve the radiation hardened performance.

A floating-body effect in partial-depletion SOI not only induces a “kink effect” but also triggers the parasitical BJT effect when lots of transient photocurrent is generated in the radiation environment[19]. An efficient method to avoid this parasitical BJT effect is tying the bodies of SOI transistors to a stable voltage[19]. There are three types of body tied embodiment, as shown in Fig. 8: BTS-type, T-type and H-type. The stable body provides the ability to transfer transient photocurrent quickly and to improve the radiation hardened performance. Concretely, the smaller body tied resistor is better for transferring transient photocurrent quickly. So, the transfer transistor in our PP is constructed by an H-type body-tie structure instead of a T-type body-tied structure. Considering the area effect, the load transistor and drive transistor in our PP are constructed by a BTS-type body-tied structure.

4. Verification and implementation

4.1. Design verification

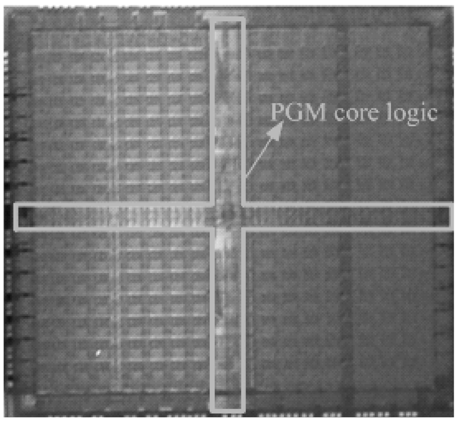
The functionality of the PGM circuit is verified in 3 levels of accuracy—behavioral-level, switch-level and mixed-level. Also, some critical paths are modeled to be verified in full transistor level.

The mixed-level simulation in which the PGM circuit is described hierarchically in both the behavioral and the transistor-levels is essential considering the trade-off between accuracy and simulation time.

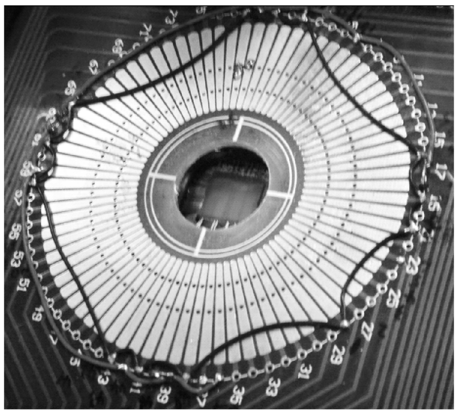
In particular, the critical path, such as the bitline and wordline, as shown in Fig. 1, should be treated specially. In order to simulate the worse case of these critical paths, we make a circuit model composed of column memory cells along the wordline, row memory cells along the bitline, wldec logic, wr logic and the computed parasitical R/C load to execute full transistor level simulation.

4.2. Layout implementation

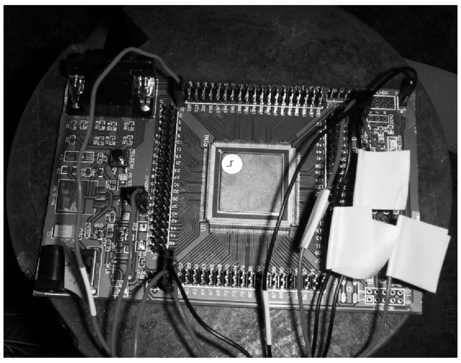
Our FPGA design is implemented with a 0.5 μm PD SOI CMOS 1P3M process. The layout of the entire chip including the PGM circuit, which is done by full-custom design. The key constraints of the pgm layout floorplan come from the require-



(a)



(b)



(c)

Fig. 9. (a) FPGA die photo. (b) FPGA die test card. (c) FPGA radiation test board.

ments of pitchmatching the wordlines and bitlines from pgm core logic with the programming point structure embedded in the logic plane. Simultaneously, hundreds of routing lines traverse the whole chip in the horizontal and vertical directions, which also constrains the PGM layout. According to the full chip floorplan, in our design we have metal M2 as the bitline in the horizontal direction and metal M3 as the wordline in the vertical direction. The programming point layout is embedded in the logic plane layout. The PGM core logic layout photo is shown in Fig. 9(a). The vertical long strip area contains the pgmctl, dispatch, wldec, blinout and gbwlout modules. The horizontal rectangular area, except for the middle cross-section, is composed of all the wlout logic (lbwlout and iobwlout) corresponding to the blocks of the individual column.

Table 1. Comparison with Spartan PGM circuit.

	Spartan PGM	Our PGM
Logic process	0.5 μm bulk-CMOS	0.5 μm SOI-CMOS
Write frequency (MHz)	12.5	25
Read frequency (MHz)	2	25
Configuration mode	3	3
Partial configuration	no	1 bit
Radiation hardened consideration	no	yes

Table 2. Radiation test results.

	Dosage	Result
TID	1×10^5 rad(Si)	no failed
Dose rate	1.5×10^{11} rad(Si)/s	no failed
Neutron fluence	1×10^{14} n/cm^2	no failed

4.3. Function test result

After the FPGA was taped out, we developed a die test card, as shown in Fig. 9(b). The fabricated chip was tested on HILEVEL automatic test equipment. The bitstream generated by our software was successfully downloaded into the FPGA and correctly read back. Because Xilinx Spartan series FPGAs are also based on the 0.5 μm CMOS logic process, we made a comparison of the configuration aspect. Table 1 provides the comparison results.

4.4. Radiation test result

Figure 9(c) is our radiation test board. Our FPGA is the only active device in this test board. We have three kinds of radiation test: total ionizing dose (TID) test, dose rate test and neutron fluence test. In the TID test, the bitstream is downloaded and read back every 5 min as the total dose increases. In the dose rate test, the bitstream is downloaded first before radiation and then read back after radiation. During the neutron fluence test, the bitstream is also downloaded and read back every 5 min as the neutron fluence increases. In these tests, the FPGA is considered as failed if we detect an error in read back bitstream. Table 2 lists the radiation test results.

5. Conclusion

A novel programming circuit has been designed and implemented for our SOI-SRAM-based FPGA chip. It provides the ability to write user-defined configuration data into an FPGA and read back this data from the FPGA with three modes. The proposed direct-access programming point scheme and partial configuration control register provide a flexible configuration option, which can easily resolve the memory coherence problem. The hierarchical simulation scheme will trade off the time overhead and accuracy. Both of the radiation hardened by technology (RHBT) and design (RHBD) are used to improve the radiation-hardened performance of it. The radiation test results show that the programming circuit has total dose tolerance of 1×10^5 rad(Si), dose rate survivability of 1.5×10^{11} rad(Si)/s and neutron fluence immunity of 1×10^{14} n/cm^2.

References

[1] http: //www.xilinx.com
[2] http: //www.altera.com
[3] Dangla D, Bancelin B. FPGA for space. 2010
[4] Wick D G. Honeywell International Advanced ASICs for Obsolete FPGA Replacement. 2007
[5] Kuboyama S. Development status for JAXA critical parts. 2009
[6] Bernstein K, Rohrer N J. SOI circuit design concepts. New York: Kluwer Academic Publishers, 2002
[7] Schwank J R, Ferlet-Cavrois V, Shaneyfelt M R. Radiation effects in SOI technologies. IEEE Trans Nucl Sci, 2003, 50(3): 522
[8] Huang W J, McCluskey E J. A memory coherence technique for online transient error recovery of FPGA configuration. Proc FPGA, 2001
[9] Calson I, Andersson S, Natarajan S, et al. A high density, low leakage, 5T SRAM for embedded caches. Solid-State Circuits Conference, 2004: 215
[10] Xilinx Inc. Spartan and Spartan-XL Families Field Programmable Gate Arrays. Jun, 2002
[11] Xilinx Inc. Spartan-3 FPGA Families: Complete Data Sheet. April, 2006
[12] Xilinx Inc. Virtex Series Configuration Architecture User Guider. Oct, 2004
[13] Xilinx Inc. Spartan-3 Advance Configuration Architecture. Dec, 2004
[14] Schultz D P, Hung L C, Goetting E E. Method and structure of configuring FPGAs. USA Patent, No. 6204687 B1, Mar, 2001
[15] Wang J, Chen L G, Lai J. FPGA downloading circuit design and implementation. The 8th Int Conf on Solid-State and Integrated-Circuit Technology, 2006: 1950
[16] Carmichael C, Fuller E, Blain P, et al. SEU mitigation techniques for Virtex FPGAs in space applications. MAPLD, 1999
[17] Xing K F, Yang J, Wang Y K, et al. Study on the anti-radiation technique for Xilinx SRAM-based FPGA. Journal of Astronautic, 2007, 28(1): 123
[18] Calin T, Nicolaidis M, Velazco R. Upset hardened memory design for submicron CMOS technology. IEEE Trans Nucl Sci, 1996, 43(6): 2874
[19] Colinge J P. Silicon-on-insulator technology. Boston, Massachusetts, USA: Kluwer Academic Publishers, 1993