Embedded SRAM Initialization Using External Serial EEPROM

Table of Contents

Introduction

Embedded SRAM blocks have become common in FPGA design. Since SRAM is a volatile memory type, the stored data vanishes in the absence of power. When power is restored, the memory is empty. As many applications operate on data stored in SRAM, it must be filled, or initialized, on power-up.

Microsemi IGLOO®, ProASIC®3, Axcelerator, ProASICPLUS, and ProASIC FPGA families of devices support embedded SRAM blocks. One method of initializing the on-chip SRAM blocks is to store the initialization data in an external non-volatile EEPROM and transfer it to the internal SRAM blocks at power-up. This document offers an efficient and low-cost solution for implementing this initialization method. Figure 1 on page 4 shows a block diagram of the RAM initialization using external serial EEPROM. A reference design is provided that can be used (shown as I2C Interface in Figure 1 on page 4) in all Microsemi FPGAs that contain embedded memory blocks. The reference design is simulated, and the simulation files are also included in this application note.

Embedded SRAM Blocks in Microsemi FPGAs

Table 1 lists several features of the SRAM blocks in various FPGA devices and clarifies their differences.

Table 1 • **FPGA Embedded SRAM Features**

FPGA Family	Maximum SRAM Bits	Variable-Aspect Ratio	True Dual-Port
IGLOO	144k	Yes	Yes
IGLOOe	504k	Yes	Yes
IGLOO PLUS	36k	Yes	Yes
ProASIC3E	504k	Yes	Yes
ProASIC3	144k	Yes	Yes
Axcelerator	288k	Yes	No
ProASIC PLUS	198k	No	No
ProASIC	63k	No	No

Note: For more information on embedded SRAM blocks, refer to each FPGA datasheet located on the Microsemi website (www.Microsemi.com).

As illustrated in Table 1, there are variations in size and features of memory blocks for different FPGA families. Although these variations may require changes for a specific implementation, they are not significant enough to affect the fundamentals of the reference design. Therefore, a single reference design targeting ProASIC[PLUS] FPGAs is presented in this document. The effects of feature and size variations are discussed in the "Instantiation of the Reference I2C Interface in the User Design" section on page 5.

Serial EEPROM Devices

A serial EEPROM is used as the source of the initialization data for the following reasons: low cost and small footprint. There are many vendors providing different types of serial EEPROM devices; however, the majority of them have similar pins and functionality. The reference design in this application note connects to serial EEPROM devices from Atmel that use the I^2C interface protocol.

A serial EEPROM is usually configured into multiple pages of multiple bytes. For example, an Atmel AT24C02A device contains 2k memory bits that are configured into 32 pages of eight bytes each. Therefore, eight bits of address are required to address a particular memory location (containing a data byte) within this device.

Complete information about pin configuration, functionality, operating condition, and electrical characteristics of Atmel Serial EEPROM devices is available at Atmel (www.atmel.com) or any other serial EEPROM vendor's website. The datasheet for the Atmel AT24C02A serial EEPROM used in this application note is *Two-Wire Serial EEPROM AT24C02A/04A/08A/16A*.

RAM Initialization Reference Design

The reference design is described and analyzed in three sections. The first section discusses the functionality, architecture, and operation basics of the design. The second section presents a VHDL code that implements the RAM initialization design and demonstrates the functionality of the code by illustrating the simulation results. The final section provides guidelines on how to instantiate, how to use the reference design in user design, and how to connect the FPGA to the serial EEPROM.

Architecture and Functionality

This design implements an I^2C master interface that will read data from an external EEPROM (AT24C02A) at power-up to initialize a 256x8 internal SRAM block. When external logic signals for a Non-Volatile Memory (NVM) write access, the block reads data from the external source and writes it to the specified NVM address.

Clock Management

The reference design runs on a main clock input named CLK. As described in the AT24C02A datasheet, the transitions on the SDA line of an EEPROM are only allowed while the SCK input is low; otherwise it will cause a start or stop condition. Therefore, CLK is divided by two to provide an initialization clock. This clock is called BTCK. The clock division allows the design to run on the fast clock (CLK) and data transfer is easily coordinated with the active edges of the initialization clock (BTCK). Two clock outputs of the reference design (ICLK and SCK) are derived from BTCK. SCK is the clock input to the serial EEPROM. SCK follows BTCK except when the reference design is generating a start or stop condition. In these cases, the SCK is kept high so that the transitions on the data line (SDA input to AT24C02A) can enter the EEPROM in the start or stop condition. ICLK is the embedded SRAM initialization clock. Writing into the SRAM is synchronized with the rising edge ICLK. ICLK is an inverted version of BTCK since the data transmission from external EEPROM is synchronized to the falling edge of the clock.

State Machine

The heart of the design is a 9-stage state machine. The following are the state definitions:

0000 – reset state: generate a start bit and load 0xA0 command

0001 – send byte: load 0x00 address

0010 – send byte: generate a start bit and load 0xA1 command

0011 – send byte: clear byte count

0100 – receive byte: if byte count! = FF: acknowledge, count++, go to state 0100 else assert "nack"

0101 – stop: assert stop bit and loop until updt = 1; then generate a start bit and load 0xA0

0110 – send byte: send byte; if nack asserted go to 0101, else load write address

0111 – send byte: send write address, load write data

1000 – send byte: send write data and go to 0101

The state machine implemented in this design is actually a counter that starts at zero and counts up, then jumps back to 0101 and counts up again, returning to zero only when global reset is asserted low.

Counters

The reference design tracks the number of bits and words received from external EEPROM using two counters: BCNT and CCNT. BCNT is a 4-bit counter that increments in state 0100 (reading from RAM) with the falling edge of CLK whenever BTCK is low. MSB of BCNT reaching a value of one indicates transfer of eight bits. The counter is cleared whenever its MSB reaches a value of one. It then starts counting for the next data word. The CCNT counts the number of data words received during power-up initialization (from 0x00 to 0xFF in this example). The CCNT output is also used as the embedded SRAM write address.

SRAM Interface Ports

Embedded SRAM blocks of the FPGA connect to the I^2C interface design through the following ports:

- IENB: An output of the I^2C interface, which acts as a write enable to the SRAM. This signal is driven low (enabled) by the interface during each acknowledge step after receiving the eight bits of data (one byte). This is because the embedded SRAM in this reference design is configured as a 256x8 block.
- IADDR: Directly extracted from CCNT output, connects to the write address of the SRAM block
- IDATA: Write data input to the SRAM.
- ICLK: The SRAM initialization clock

Acknowledge

Acknowledge is a handshaking process between the EEPROM and FPGA in different operations. During the data read (initialization), the FPGA should send a value of zero on the data bus, after receiving eight bits of data, to acknowledge the receipt of the byte. The EEPROM starts sending the next byte of data after receiving the acknowledge signal. Also after each device addressing operation or write operation, the EEPROM sends out a value of zero in order to confirm the receipt of data from the FPGA. During operations in which the I^2C interface (FPGA) needs to send the acknowledge signal (e.g., read), the reference design places a value of zero on the data line (SDO) automatically after receiving eight bits of data (checking BCNT). When the acknowledge signal is sent by EEPROM (e.g., device addressing), the I^2C checks the data bus (SDI) after receiving eight bits (ninth bit). If the data line is high during the ninth bit, I^2C activates the "not acknowledged" flag (NACK). If the state machine is in states 0000 to 0011, assertion of NACK will cause the state machine to go back to 0000. In states 0100 or higher, the assertion of NACK will cause the state machine to go to state 0101 (STOP) and wait for UPDT or RESET.

Status Output

INIT output of the reference design indicates the state of I^2C. At power-up reset, it is asserted high to indicate the start of initialization process. It remains high until the state machine enters state 0101 (STOP). This will indicate the end of the initialization process and that the interface is available to write into the EEPROM.

Writing into the EEPROM

The I^2C initialization interface design features byte-write into the EEPROM. The reference design assumes that the data to be written into the EEPROM is supplied by an external FIFO; however, it can be imported from any source within the user design. The write process starts once INIT is low (state machine is in state 0101) and the UPDT input is asserted high. UADDR and UDATA are address and data inputs to be sent to the EEPROM. The UENB output is used as an active low read enable to the external logic (e.g., FIFO). UENB becomes active low in state 0111 when the I^2C interface is finishing sending the write address to the EEPROM. The user's design should clear the UPDT control unless another address and data byte are ready to be written into the EEPROM.

I²C Operation Flow Diagram

Figure 1 illustrates the flow diagram of the I²C interface design and summarizes all the procedures and functionalities explained above.

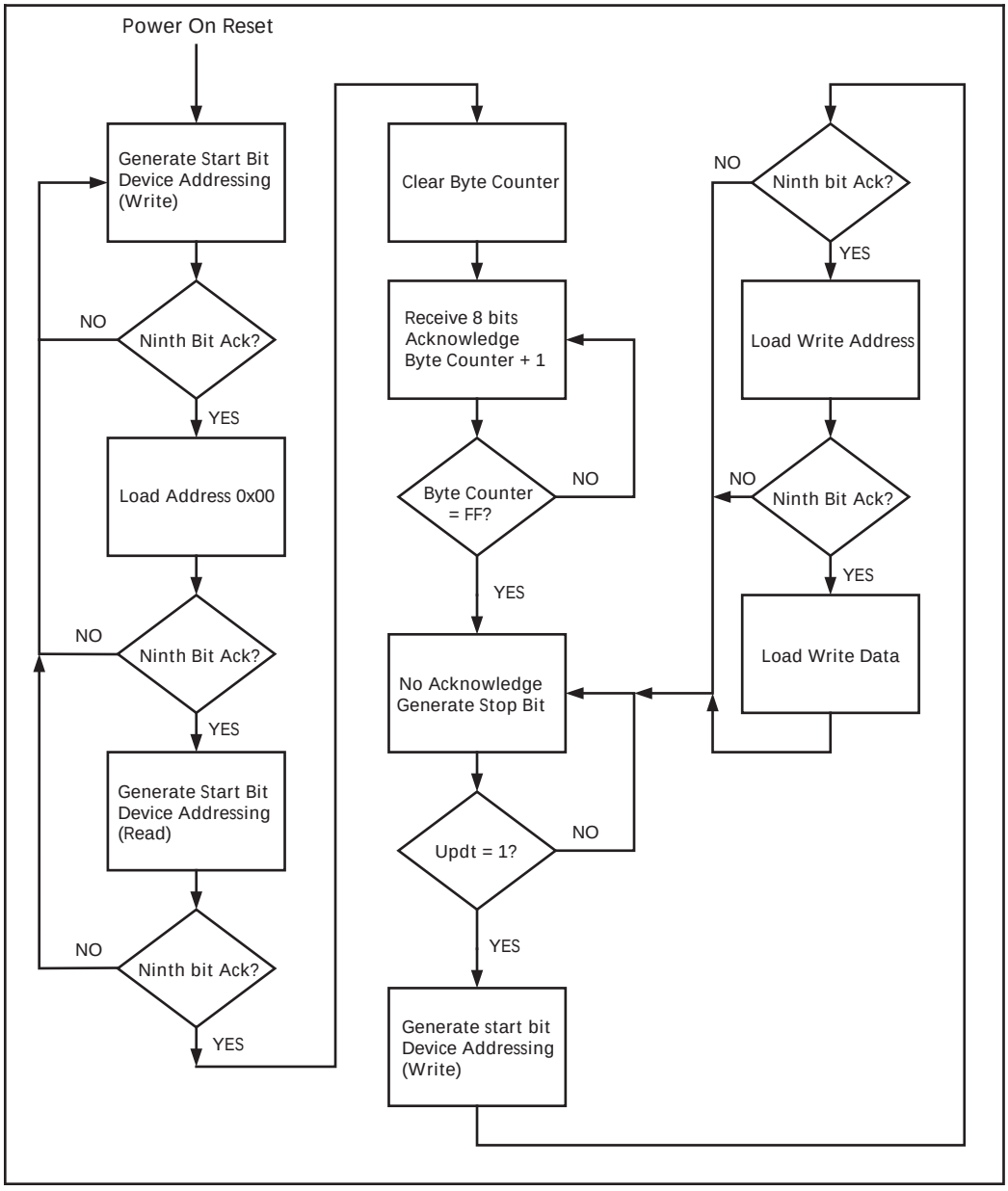

Figure 1 • **I²C Interface Design Flow Diagram**

I²C Interface HDL Source

The design files on the Microsemi website provide both the VHDL and Verilog versions of the code for the I²C interface reference design, called eeprom_interface. As mentioned earlier, the eeprom_interface design targets an Atmel AT24C02A device and initializes a 256x8 embedded SRAM block in the FPGA.

Table 2 shows the utilization of the eeprom_interface design on three different Microsemi FPGAs.

Table 2 • **Utilization of I²C Reference Design in Microsemi FPGAs**

Device	AX250	APA150	A3P400
Utilization	2.89%	3.2%	1.94%

A simple model[1] has been developed to partially simulate the AT24C02A serial EEPROM behavior. This model, along with a testbench, accompanies the eeprom_interface design for simulation purposes.

Instantiation of the Reference I²C Interface in the User Design

The eeprom_interface design presented in the previous section is used along with the rest of the user design. Therefore, it should be instantiated in the top-level of the user's design, connecting to embedded SRAM blocks. This section provides guidelines on how to use eeprom_interface in an upper level design.

Connection to External EEPROM

The FPGA connects to the serial EEPROM via two main ports: SDI and SCK. Use of WP is optional; it can be tied to ground. Please refer to *Two-Wire Serial EEPROM AT24C02A/04A/08A/16A* for more information. The SCK input of the serial EEPROM connects directly to the SCK output of the eeprom_interface design. The I/O standard used in the FPGA depends on the V_{CC} value of the EEPROM device:

V_{IL}: Between -0.6V to 0.3xVCC

V_{IH}: Between 0.7xVCC to VCC + 0.5

The external serial EEPROM uses an SDA (bidirectional line) for data. The eeprom_interface has two ports, one for each direction: SDI and SDO. The top-level design in the FPGA should connect these two ports to the SDA of the EEPROM. The top-level of the user's design must create an open-collector bidirectional driver to connect to SDA. This is done using the following statements in the top-level HDL:

```
SDI <= SDA;
SDA <= '0' when (SDO = '0') else 'Z';
```

where SDA is the top-level port of the design connecting to the serial EEPROM.

In addition, the SDA line on the board must be pulled up to V_{CC}. A high impedance value (Z) in the second statement should be modified to high (H) during HDL simulation of the design to resemble the external pull-up on the line.

Connection to Embedded SRAM Blocks

The reference design connects to the internal embedded SRAM via four ports: IENB, IADDR, IDATA, and ICLK. The functionality of these ports is defined in previous sections. IGLOO and ProASIC3 families of devices offer true dual- port embedded SRAM blocks. Therefore, eeprom_interface SRAM interface ports can connect to one of the ports while the user design accesses the other. The embedded SRAM blocks in Axcelerator and ProASIC^{PLUS} FPGAs are two-port memory blocks offering one read and one write port. The eeprom_interface should connect to the write port of SRAM during initialization. If the user's design needs to access the write port of the embedded SRAM block, a simple MUX arbiter should be implemented, as shown in Figure 2 on page 6.

If designers intend to build dual-port RAM blocks in ProASIC^{PLUS} or Axcelerator FPGAs using the embedded SRAM blocks, guidelines are given in the following application notes located on the Microsemi website:

1. *The AT24C02A model offered with this application note is not a complete model of this device. This model has not been certified by Atmel, Microsemi, or any other vendor. This simulation model is intended to be used only to verify the functionality of the I²C reference design in this application note.*

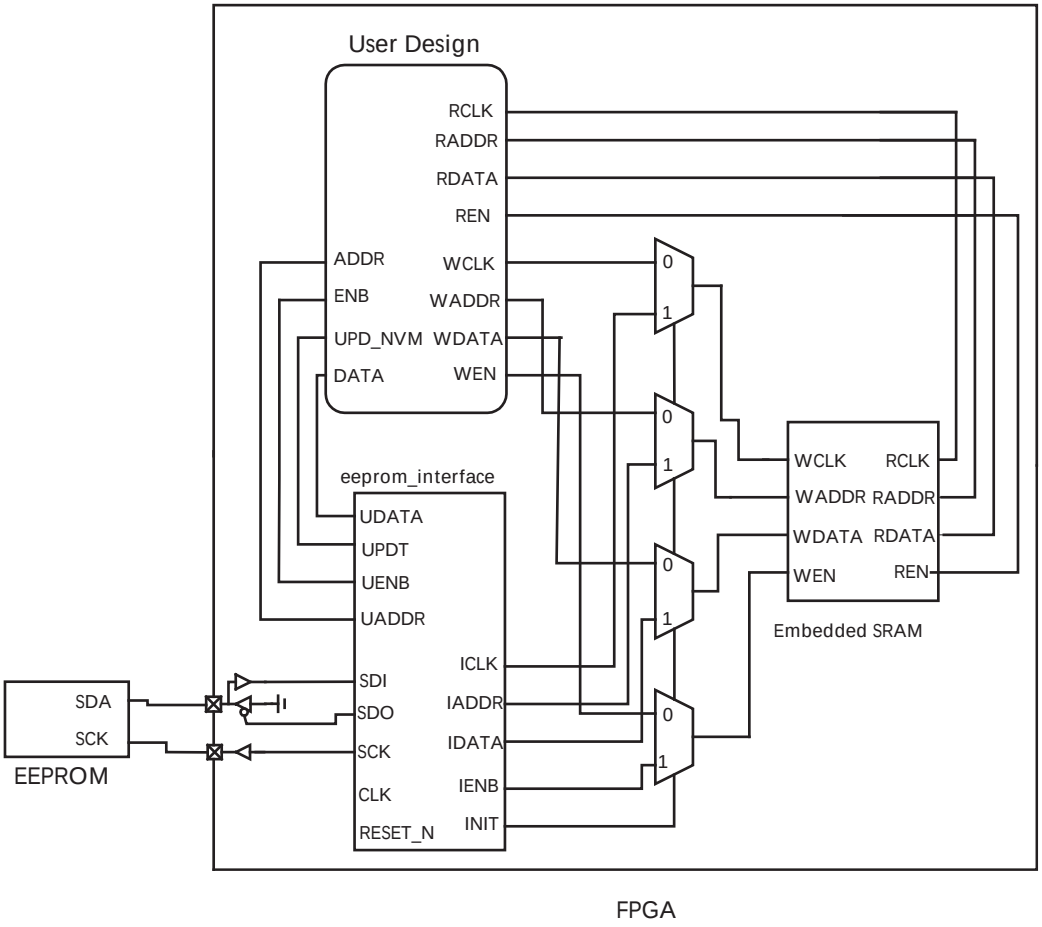

Figure 2 • **MUX Arbiter To Access Two-Port RAM**

Implementing Multi-Port Memories in ProASICPLUS Devices and *Implementing Multi-Port Memories in Axcelerator Devices*.

Timing Requirements

The commercial two-wire serial EEPROMs available in the market run on very low clock speeds (a few hundred kilohertz to one megahertz). Therefore, the I^2C interface design does not require strict timing constraints to meet the required initialization speed. However, if the clock signals are not routed through global networks they may be prone to a considerable amount of skew. Users should check for possibility of hold time violations in the presence of clock skew.

Interfacing SRAM Blocks of Different Depth or Width

The target of the eeprom_interface design presented in this application note is a 256x8 SRAM block. If the memory blocks in the user's design are different, minor adjustments are required in the reference design.

Depth

If the depth of the SRAM block is different from 256, the size of the CCNT counter and IADDR must be modified accordingly.

Width

The data transaction on the EEPROM side of the interface is consistently done in 9-bit (8-bits of data and 1-bit acknowledge) packets. Therefore, the size of BCNT should not be modified. SRAM blocks in Axcelerator, ProASIC3, and IGLOO families of devices offer a variable-aspect ratio. This feature facilitates the memory initialization when the memory width is other than 8-bit. If this feature is used, the write port of the RAM should be set to a width of eight while the read port is configured according to the user's design requirements. However,

if the variable-aspect ratio feature is not used or the target FPGA is a ProASIC^{PLUS} device, then the eeprom_interface design should be modified accordingly. For example, if the target SRAM is configured as a 256x16 block, the following modifications/adjustments will be required in order to perform a complete initialization:

1. The external EEPROM should be at least 4k (512x8). Two consecutive memory locations store one initialization word (16-bit). The first byte stores the upper half of the word and the second byte stores the lower half.
2. Add an 8-bit MSB_REG register to the I^2C code.
3. Set CCNT for the number of BYTES (in this example the CCNT should be nine bits wide).
4. Generate two write enables; the first one writes into the MSB_REG when CCNT(0) = 0, the second one uses MSB_REG and SDATA (concatenation) to write into the memory at 16-bits wide when CCNT(0) = 1.
5. With the above modifications, IENB and WRI can be removed from the I^2C reference design code.

Conclusion

Volatile SRAM blocks, embedded within FPGAs, can be initialized after power-up using an external serial EEPROM. This example application uses a minimum of two FPGA user I/Os to interface with the external EEPROM. The external memory can also be updated (written into) by the FPGA if needed. This application note presents an interface that can be instantiated into the user's design, performing the initialization at power-up. The reference design utilizes a very small portion of the FPGA logic for implementation and does not affect the performance of the main design. The design in this document initializes a 256x8 SRAM block but can be easily modified to support memory organizations of different width and depth.

References

Two-Wire Serial EEPROM AT24C02A/04A/08A/16A

http://www.atmel.com/dyn/resources/prod_documents/doc0976.pdf

Implementing Multi-Port Memories in ProASICPLUS Devices

Implementing Multi-Port Memories in Axcelerator Devices

List of Changes

The following table lists critical changes that were made in each revision of the document.

Revision	Changes	Page
Revision 2 (March 2015)	Non-Technical Updates.	N/A
Revision 1 (November 2008)	First Release.	N/A

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA

Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996

E-mail: sales.support@microsemi.com

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense & security, aerospace and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; security technologies and scalable anti-tamper products; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif., and has approximately 3,400 employees globally. Learn more at **www.microsemi.com**.

51900081-2/03.15