HB0846

CoreBootStrap v3.0Handbook

October 2018

Microsemi

a Microchip company

1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

1.1 Revision 1.0

Revision 1.0 is the first publication of this document. Created for CoreBootStrap v3.0.

- PolarFire support added
- Removal of support for Big Endian
- SPI source address and AHB destination address split into upper and lower words
- Updated figure for system integration

Contents

Figures

Tables

2 Introduction

2.1 Overview

CoreBootStrap connects as a master to a 32-bit AHB bus on which an on-chip RAM block, that can also be accessed by the processor resides. CoreBootStrap allows booting a soft processor from a SPI flash device indirectly by first copying the boot code from SPI to this RAM, then allows the processor to boot from there.

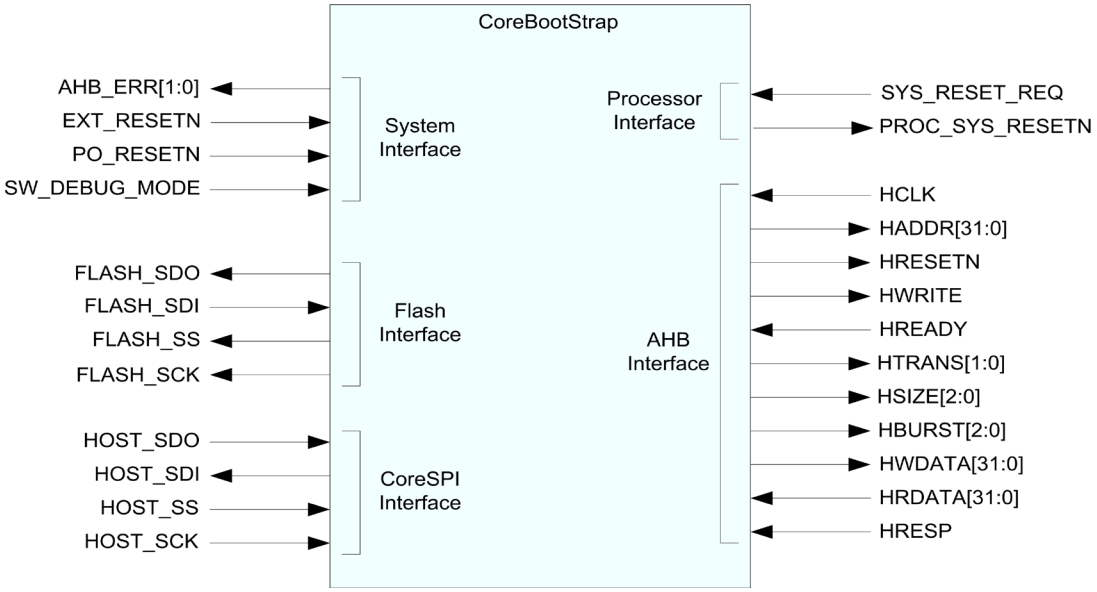
Following assertion of one of a number of reset sources, the core asserts the reset line of the soft processor, holding it in reset as it copies the processor's executable program from an external SPI flash device into the processor's instruction memory space. Assuming there are no errors, it then releases the processor's reset line, which allows the processor to boot directly from on-chip memory.

Note: If an error occurs, then reset will not be released.

Having initially taken ownership of the SPI interface, the core finishes by re-connecting it to the processor, which typically accesses it thereafter through an APB-based CoreSPI core.

CoreBootStrap uses **Motorola Mode 0** signalling and as such will work with all current SPI flash devices. It is parameterized to handle both small and large devices by selecting three or four byte addressing, to facilitate use with all SPI chip sizes.

Figure 1 Input / Output Signal Diagram



2.2 Features

- Provides bootstrap capability for a processor sharing an AHB bus, whereby boot code is extracted from SPI and placed in AHB-based RAM for fast execution after exiting the reset state
- Supports all available SPI Flash chips, through **Motorola Mode 0** signaling, and parameterized software reset command sequences along with various timing parameters to handle differences between SPI chip manufacturers
- Three and four byte addressing options parameterized to handle SPI flash chips of different sizes
- Supports three reset sources:
    - Power-on reset
    - External reset
    - Processor reset
- Resets extended to parameterized durations:
    - Power-on reset length to overcome SPI Flash chip's widely-varying specifications from power-up to device available, which can range from under 200us to over 5ms
    - Default reset duration parameterized separately to cover the other two reset sources
- SPI Flash source and AHB RAM destination addresses are parameterized along with the 32-bit word count for the boot code copy operation
- AHB master-mode compatible, including handling AHB "response" in case of error conditions
- Posted-write protection through read-back of first location in AHB memory at end of copy
- Safe transfer of SPI bus to host after boot via Flash Slave-select (FLASH_SS) glitch avoidance
- Software debug support: bypassing of boot code copying and error signals provided
- SPI clock programmable as any even-number ratio of AHB clock to SPI clock, minimum ratio 4:1
- Support for VHDL and Verilog netlists

2.3 Limitations

- AHB bus sizes other than 32-bits are not supported
- SPI flash and AHB RAM start addresses must start on 4-byte word boundaries
- Boot code must be a multiple of 4 bytes
- No support for big endian

2.4 Core Version

This handbook supports CoreBootStrap version 3.0.

2.5 Supported Families

- SmartFusion®2
- IGLOO®2
- RTG4™
- PolarFire®

2.6 Supported Tools

CoreBootStrap is supported by Libero SoC V11.8 and later and Libero PolarFire v2.0 and later.

2.7 Supported Interfaces

- AHB and AHB-Lite 32-bit master interface. For more information, refer to: AMBA® 3 AHB-Lite Protocol
- SPI serial flash interfaces, which can handle **Motorola Mode 0** signaling (all available SPI Flash chips support this mode). For more information, refer to: Micron Technology N25Q00AA Specification (1Gbit SPI Flash chip)
- SPI host interfaces, for more information, refer to: Microsemi “CoreSPI_HB.pdf”

2.8 Device Utilization and Performance

CoreBootStrap has been implemented in several devices of Microsemi using standard speed grades. Table 1 lists a summary of various implementations data.

Table 1 Resource, Utilization, and Performance Details

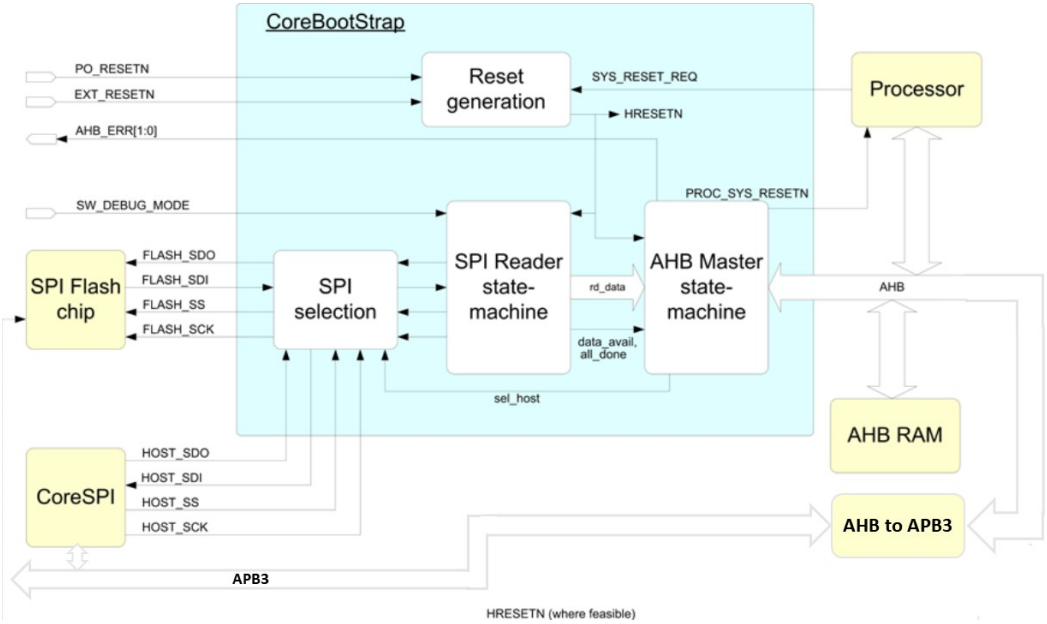
Family	Cells or Tiles			Utilization		Performance (MHz)
	Sequential	Combinatorial	Total	Device	Total (%)	
RTG4	481	332	813	151824	0.54	108
IGLOO2	478	332	810	27696	2.92	166
SmartFusion2	478	332	810	86184	0.95	166
PolarFire	543	343	886	299544	0.29	200

Note: Data in this table was gathered using typical synthesis and layout setting on standard parts and speed grades (PolarFire at -1 speed grade) at standard temperatures.

3 System Overview

CoreBootStrap provides a Master interface to the system AHB or AHB-Lite bus, on which an AHB RAM block resides, and to which a processor has access when booting. It also connects to a SPI Flash chip, and to a host SPI controller. It has direct processor reset request inputs and provides a processor reset output, which it releases on completion of the boot process. For debug option, it has a debug mode input, which bypasses the boot copy, and error output signals are provided. Figure 2 shows the top-level block diagram.

Figure 2 Top-level Block Diagram



4 Functional Description

This section describes the components of the CoreBootStrap controller and how the controller operates.

The sequence of events is as the following:

Boot reset -> SPI Control -> AHB control -> SPI Selection.

Firstly, the Boot Reset generator acts in asserting HRESETN once any of the reset inputs assert. When reset timing is complete, it removes HRESETN.

Next SPI Control initializes SPI Flash control registers and extracts a 32-bit word, indicating to AHB that one word is ready.

Then AHB Control writes this to AHB memory and waits for the next word from SPI. As the HCLK frequency is a multiple of that of the SPI clock, and it takes at least 64 SPI clocks to read one 32-bit word, and the AHB controller owns the AHB bus, it is guaranteed that the 32-bit word will complete its transfer to AHB memory well before the next word arrives from the SPI control block. Therefore, no FIFO buffering or handshake scheme is required between SPI and AHB.

The SPI controller keeps track of the amount of data copied and after the last one is transferred it indicates this to the AHB interface.

The AHB interface then does a posted-write check, and points the SPI Selection multiplexer back to the Host SPI interface, and finishes by removing the reset from the processor. If an error occurs during the transfer of the AHB_ERR signal will be asserted.

The following sections describe these steps in more detail, also explaining how to set-up the various parameters to work with the selected SPI Flash chip manufacturer and chip size.

4.1 Source and Destination Address Concatenation

The SPI source address and destination address are configurable in their upper and lower words, they will be concatenated together when the core is running.

4.2 Reset Generation

The SPI source address and destination address are configurable in upper and lower words, The request input from the processor, SYS_RESET_REQ is inverted to produce an active low version, **SYS_RESET_REQN**.

Three active low reset sources are then available:

PO_RESETN: power-on reset, active low

EXT_RESETN: external reset, active low

SYS_RESET_REQN: processor reset request, active low

These are combined through an AND gate to provide one **async_resetn** asynchronous reset signal, which forms the starting point of the reset sequence.

In addition, PO_RESETN is latched to create an internal flag **rst_by_por** as it is treated differently in regard to the length of the reset output signal, HRESETN.

Once **async_resetn** asserts, HRESETN asserts asynchronously. When **async_resetn** de-asserts a counter is released, counting to a parameterized reset duration (that is, measured in HCLK periods), thereafter, releasing RESETN synchronously. If the **rst_by_por** flag is set, then HRESETN is held asserted from **async_resetn** deassertion for **RST_POR_DURATION** HCLK cycles, whereas it is held for **RST_EXTPROC_DURATION** HCLK cycles otherwise.

The following considerations are used in determining the values for these two parameters.

4.2.1 RST_POR_DURATION

After power-on, all SPI Flash manufacturers specify the time from the power-rail (VCC) reaching minimum operating voltage and the SPI chip being ready for chip-select to assert for polling purposes or control register writes (not data writes). On Micron parts for example, this is “VTP” (VCC to polling time) and must be at least 150µs. On Winbond parts this is called “tVSL” (VCC to Select) and is specified as 20µs minimum. ISSI chips specify it as “tVCE” (VCC to Chip Enable) with a specification of 1ms, while Macronix also call it tVSL but with a specification of 3ms. For Spansion/Cypress it’s “tPU” (power up) with a spec of 300µs; GigaDevice specifies it as tVSL specified as 10µs, and for Adesto it’s tVCSL with a specification of 70µs minimum.

The Reset block holds reset active for this time to ensure no SPI Flash accesses are attempted too soon after power-up. It may be that the board-level hardware or other on-chip power-on reset conditioning has taken place already with such times accounted-for, in which case RST_POR_DURATION can be set to a lower value.

4.2.2 RST_EXTPROC_DURATION

If HRESETN is connected to the RESET# of the SPI Flash chip, then the duration of reset must exceed the minimum hardware reset pulse specification for the SPI chip. For Micron this is just 50ns; Spansion/Cypress is 200ns; for ISSI, GigaDevice and Winbond it is 1µs; for Macronix and Adesto it is 10µs.

If HRESETN is not driving the SPI chip’s reset pin, then it need only be set to a few HCLK cycles, such as four.

4.3 SPI Control

corebootstrap_spi_reader.v contains two modules: corebootstrap_spi_ctrl.v, which implements the SPI controller function, and corebootstrap_cksum_ctrl.v, which is a placeholder block for a future implementation of a data-check function.

The state-machine in corebootstrap_spi_ctrl.v starts once HRESETN is released. It performs the following sequence of operations:

1. **Hardware Reset Recovery**. After release of HRESETN, the state-machine waits a period of "RST_RECOVERY_DURATION" HLK cycles to ensure internal reset actions in the SPI Flash chip are completed. For Micron this is 40ns; Spansion/Cypress is 200ns; for ISSI however it is 100µs; GigaDevice is 60µs; Winbond does not specify it; for Macronix it is 10µs, and for Adesto it is 1µs.
2. **Check for Flash Busy**. It then polls the SPI Flash's Status Register's Busy bit. This handles the case of a reset occurring during a SPI program, erase, or write to certain registers, which take time to complete.
3. **Apply a Software Reset**. The next step is to apply a software reset, if the SPI Flash chip supports it, to clear any volatile registers, which may have changed modes of operation such as addressing modes. The parameter SW_RESET_TYPE indicates which reset type applies to the SPI Flash chip.
4. For Micron, ISSI, Winbond, Macronix, and GigaDevice this is done with a 66H 8-bit command, then Slave-select de-selection for SS_DESELECT_DURATION, followed by a 99h 8-bit command. For Micron SS_DESELECT_DURATION is 40ns, ISSI just 7ns (tCEH), for Winbond it is 50ns (tCSH), for Macronix it is 30ns, and GigaDevice 20ns.
5. For Adesto software reset is done by a 32-bit command with the code "f0_00_00_00", while for Cypress/Spansion it is an 8-bit code "f0".
6. **Software Reset Recovery**. After applying Software Reset the state-machine again waits a period of "RST_RECOVERY_DURATION" HLK cycles to ensure internal reset actions in the SPI Flash chip are completed.
7. **Read data and pass to AHB**. The Flash is now ready for reads. For Flash chips of 128Mbit and above, four-byte addressing is used via the 13h command, otherwise it uses three-byte addressing via the 03h command. This is setup via the READ_4BYTE_ADDR parameter.

An inner loop fetches 32 bits of data one bit at a time, after which it removes SS (Slave Select) for SS_DESELECT_DURATION. Timings for these are listed in step 4 above. In addition for Adesto this is 30ns and for Spansion/Cypress it's 10ns. This 32-bit "rd_data" value is then passed to the AHB controller with the indication "rd_data_valid".

An outer loop increments a word counter and its SPI address, repeating step 5 until DATA_WORD_CNT words have been transferred to AHB, and completes the copy process by asserting "rd_all_done" to AHB. It also indicates to the CKSUM_CTRL block that it can perform a data check.

During the transfer the current SPI address is compared with CKSUM_SPI_ADDR, and the data at this address is latched. CKSUM_SPI_ADDR must reside at some location within the code being copied.

When the corebootstrap_cksum_ctrl.v block completes the data check it indicates the check is complete to the AHB controller, also indicating if there is been an error. In the initial release data checking is not provided, and this block simply assigns CKSUM_ERR to 0, and cksum_done to one.

4.4 AHB Control

corebootstrap_ahb_writer.v contains a state-machine which interfaces to the AHB bus, which operates as follows:

1. **Copy to AHB memory**. After HRESETN is released, it waits for the SPI control block to indicate that a 32-bit word is available to write to AHB memory, and proceeds with an AHB "NONSEQ" write cycle.

   If it is the first transfer after release of HRESETN, it latches this 32-bit word for use in comparison with read-back of that word later.

   If an error response is returned by the AHB, it sets AHB_ERR[0] and immediately stops operations, moving to a "Finish" state where it loops awaiting the next HRESETN assertion. In this case it doesn't remove reset from the processor.

   Assuming there is no AHB error, it continues with step 1 until the SPI control block indicates "rd_all_done".
2. **Wait for data checking complete**. It waits for "cksum_done" and moves on.
3. **Posted-write protection**. To ensure that all write operations were actually completely written to memory, that is, . no "posted writes" still in progress, the state-machine reads back the first location written. When AHB indicates "HREADY" for this read cycle, it implies that there are no outstanding writes.

   As an additional check, the data read from this location is compared with the value originally written, and if there is a mis-compare AHB_ERR[1] flag is set.
4. **Remove processor reset and finish.** If there are no errors, either CKSUM_ERR or AHB_ERR[1:0], the state-machine first sets "sel_host" which is used by the SPI_SEL.v block to route the SPI signals to the Host SPI path rather than the CoreBootStrap path, and a cycle later removes reset from the processor(s) by de-asserting PROC_SYS_RESETN. It then moves to a "finish" state awaiting the next assertion of HRESETN.

   If there are errors, it does not de-assert PROC_SYS_RESETN and goes directly to the "finish" state.

4.5 SPI Selection

Selection of SPI signal path between CoreBootStrap's SPI controller and the external Host's SPI controller is handled in the corebootstrap_spi_sel.v module. This is mainly a multiplexer for the SPI signals, controlled by the "sel_host" signal from the AHB controller.

In addition, Slave Select (SS) is disabled during the selection process, only enabled one HCLK cycle after reset is removed from the processor. This window of one HCLK cycle is sufficient to avoid glitches on Slave Select. Once the SPI Flash slave is not selected during the switchover, glitches caused by multiplexing the clock and data signals are disabled from causing problems.

5 Interface

5.1 Configuration Parameters

Table 2 describes the CoreBootStrap Verilog parameters for configuring the RTL code. All parameters are integer types.

Table 2 Parameter/Generic Descriptions

Parameter Name	Valid Range	Default	Description
SPI_SRC_ADDR_1	Any 16-bit value which is a multiple of 4	0	Upper 16 bits of the SPI Source start Address: location of the first 32-bit word of boot code in SPI Flash memory.
SPI_SRC_ADDR_0	Any 16-bit value which is a multiple of 4	0	Lower 16 bits of the SPI Source start Address: location of the first 32-bit word of boot code in SPI Flash memory.
AHB_DST_ADDR_1	Any 16-bit value which is a multiple of 4	0	Upper 16 buts of the AHB Destination start Address: location where the first 32-bit word of boot code will be forwarded to in the AHBLSRAM.
AHB_DST_ADDR_0	Any 16-bit value which is a multiple of 4	0	Lower 16 bits of the AHB Destination start Address: location where the first 32-bit word of boot code will be forwarded to in the AHBLSRAM.
DATA_WORD_CNT	Any 32-bit value	2048	Number of 32-bit words to be copied.
READ_4BYTE_ADDR	0 or 1	0	0 = three-byte SPI addressing 1 = four-byte SPI addressing for SPI chips ≥ 128Mbit
SPI_CLK_RATIO	4 to 32,768 (even numbers)	4	Ratio of HCLK to SPI clock frequency.
SW_RESET_TYPE	0 to 3	1	SPI Software Reset type: 0 = No software reset 1 = Command sequence 66h, 99h (covers most devices) 2 = 4-byte command “f0,00,00,00” (Adesto devices) 3 = 1-byte “f0” command (Cypress/Spansion devices)

Parameter Name	Valid Range	Default	Description
READ_STATUS_TYPE	0 or 1	0	0 = read status command code is 05h, bit 0 is Busy, active high (all devices except Adesto) 1 = read status command code is d7h, bit 7 is ~Busy, active low (Adesto devices only).
RST_POR_DURATION	Any integer value from 32,768 upwards	32,768	Number of HCLKS to hold HRESETN active following a power-on reset, after which polling of SPI can begin. SPI chips range from under 200us to over 3ms for this value.
RST_EXTPROC_DURATION	Any 16-bit value from 4 upwards	256	Number of HCLKS to hold HRESETN active following an external or processor reset. If HRESETN drives the RESET# pin of the SPI Flash chip, set this to the minimum reset width of the SPI chip, ranging from under 50ns to at least 10µs for this value. If not connected to the SPI chip, this should be set to 4.
RST_RECOVERY_DURATION	Any 16-bit value ≥ 4	8	Number of HCLKs following a hardware or software reset before enabling polling the SPI chip. This ranges from under 50ns to over 100µs.
SS_DESELECT_DURATION	Any 32-bit value ≥ 1	8	The deselect duration in HCLKs for the SPI chip's SS ("Slave Select") pin between commands.

5.2 Ports

Table 3 describes the port signals used by CoreBootStrap

Table 3 Input / Output Signals

Signal	Port width, bits	Input / Output	Description
AHB Master signals			
HCLK	1	In	AHB system clock. Used for all CoreBootStrap functions.
HRESETN	1	Out	AHB system reset. The signal is active low. Asynchronous assertion and synchronous de-assertion.
HADDR[31:0]	32	Out	AHB address.
HWDATA[31:0]	32	Out	AHB Write Data.
HRDATA[31:0]	32	In	AHB read Data.
HWRITE	1	Out	AHB-Lite write. When high, it indicates that the current transaction is a write. When low, it indicates that the current transaction is a read.
HREADY	1	In	When high, the HREADY signal indicates that the read or write cycle is complete. If a read, HRDATA[31:0] is valid if HRESP remains 0 during the read cycle.
HTRANS[1:0]	2	Out	AHB data transfer type. Default is 00 (=IDLE) from CoreBootStrap. Set to 10 (=NONSEQ) for one cycle during the address phase of a transfer to indicate a single non-sequential data phase follows. Other codes (01 or 11) are never driven.
HSIZE[2:0]	3	Out	AHB data transfer size, that is, bus width. Always 010 from CoreBootStrap to indicate a 32-bit bus size.
HBURST[2:0]	3	Out	AHB burst. Always 000 from CoreBootStrap to indicate a single cycle burst.
HRESP	1	In	AHB response signal - normally expected to be 0. If 1 during an AHB transfer an error is indicated on AHB_ERR and the processor is not taken out of reset. **Note:** If CoreBootStrap is connected to an AHB subsystem, the HRESP[0] signal of the AHB bus should be connected to this signal. CoreBootStrap does not support 2-bit AHB error responses.
Flash SPI signals			
FLASH_SDO	1	Out	Flash Serial Data Out, to the SPI flash chip.

Signal	Port width, bits	Input / Output	Description
FLASH_SDI	1	In	Flash Serial Data In, received from the SPI flash chip.
FLASH_SS	1	Out	Flash Slave Select, active low.
FLASH_SCK	1	Out	Flash Clock.
Host SPI signals			
HOST _SDO	1	In	Host Serial Data Out to the flash chip, input from the host processor's SPI interface after boot copy.
HOST_SDI	1	Out	Host Serial Data In from flash, routed to the host processor's SPI interface after boot copy.
HOST_SS	1	In	Host SPI Flash Slave Select, active low, input from the host processor's SPI interface after boot copy.
HOST_SCK	1	In	Host SPI Flash Clock input from the host processor's SPI interface after boot copy.
Processor interface signals			
SYS_RESET_REQ	1	In	System Reset Request from the processor.
PROC_SYS_RESETN	1	Out	Processor reset, active low, sent to the processor accessing AHB.
System Interface signals			
PO_RESETN	1	In	Power-on reset, active low. Need not be synchronous to HCLK.
EXT_RESETN	1	In	External reset, active low. Need not be synchronous to HCLK.
SW_DEBUG_MODE	1	In	Control signal used to allow software debugging. It will set up CoreBootStrap to bypass the copying of the executable memory from SPI into the processor memory when the signal is high on reset. **Note:** If 1 is enabled, then the debug mode is set (active-high).
AHB_ERR[1:0]	2	Out	AHB error: AHB_ERR[0] is due to receiving a HRESP during an AHB transfer, while AHB_ERR[1] is due to a mismatch between the first data written and a read-back of this location.
CKSUM_ERR	1	Out	Unused – Output driven low.
All signals are active high (logic 1) unless otherwise noted.			

6 Tool Flow

6.1 License

No license is required for the use of this core.

6.2 Obfuscated

Complete unobfuscated RTL code is provided for the core, allowing the core to be instantiated with SmartDesign. Simulation, Synthesis, and Layout can be performed within Libero® SoC.

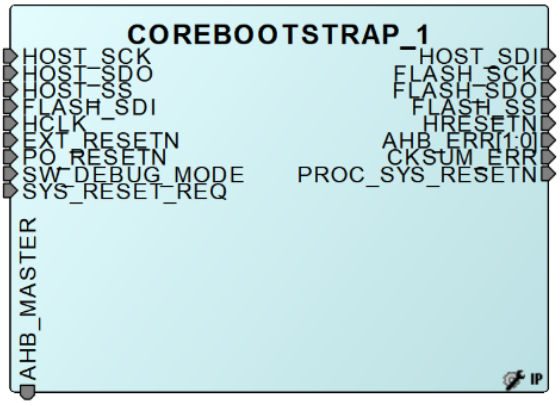
6.2.1 RTL

Complete RTL source code is provided for the core and testbenches.

6.3 SmartDesign

CoreBootStrap is pre-installed in the SmartDesign IP Deployment design environment. Figure 3 shows configuring the core using the configuration GUI within SmartDesign.

Figure 3 SmartDesign CoreBootStrap Instance View



6.4 Configuring CoreBootStrap in a SmartDesign

CoreBootStrap's parameters (listed in Table 2) can be controlled though the configuration user interface.

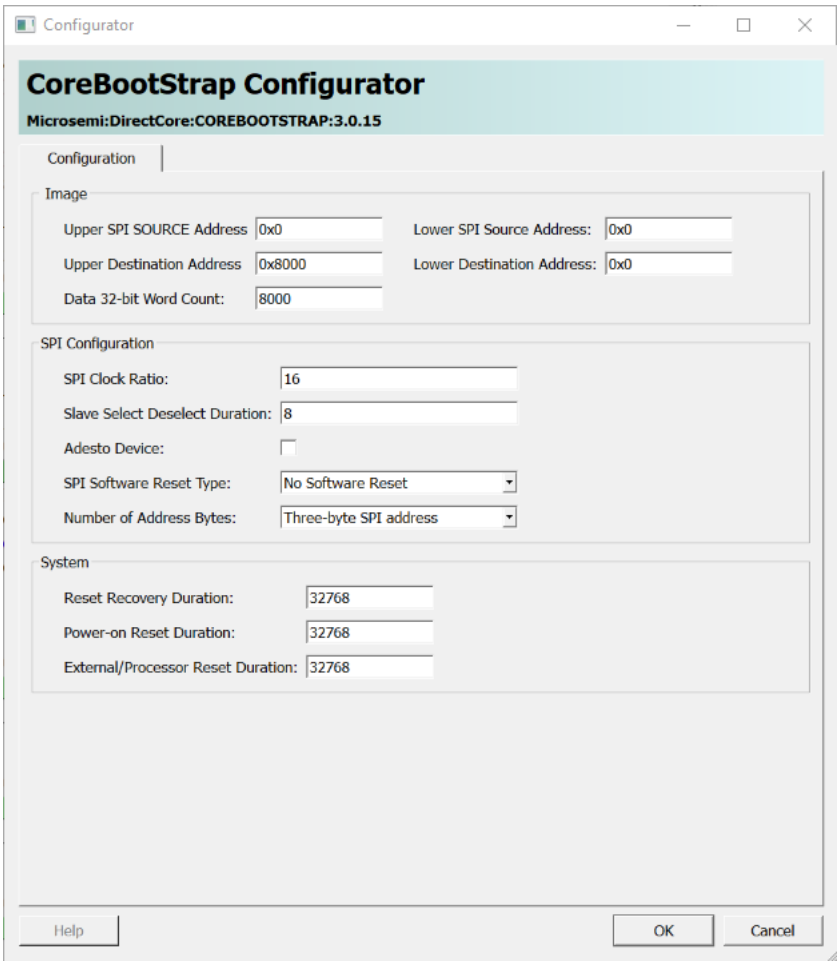
The Image section contains the parameters that control the data transfer - the SPI address, AHB destination address, and 32-bit word count.

The SPI Configuration section contains parameters relating to the SPI flash configuration needed; the clock ratio, slave select duration, whether an Adesto device is used, the reset type for the flash and the number of address bytes.

The System Configuration section configures parameters for bootstraps control over the system, such as the duration of a power on reset and how long the processor is held in reset mode.

The core is configured using the configuration GUI within SmartDesign, as shown in Figure 4.

Figure 4 CoreBootStrap SmartDesign Configuration Windows



Note: Leading zeros are suppressed, for example, 0x6000 0000 is displayed as 0x6000 0x0.

6.5 Simulation Flows

The User Testbench for CoreBootStrap is included in all releases.

To run simulations, select the **User Testbench** flow within SmartDesign and click **Save and generate** on the **Generate** pane. The User Testbench is selected through the Core Testbench Configuration GUI.

When SmartDesign generates the Libero SoC project, it installs the user testbench files.

To run the user testbench, set the design route to the CoreBootStrap instantiation in the Libero SoC design hierarchy pane and click Simulation in the Libero SoC Design Flow window. This invokes ModelSim® and automatically runs the simulation.

6.5.1 User Testbench

Figure 5 shows the structure of the CoreBootStrap simulation testbench, which includes an instance of the CoreBootStrap DUT (that is, Device Under Test), a SPI Flash chip model, a Host SPI model, AHB Memory core, an AHB monitor, and test files.

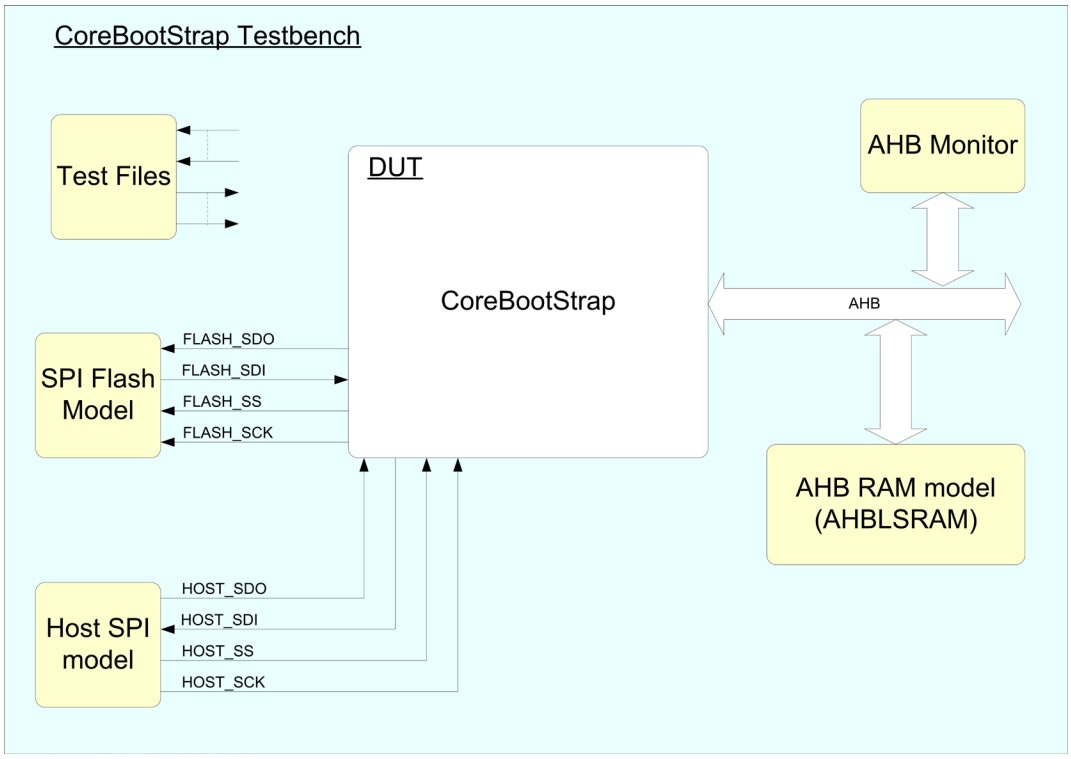
Test files setup different parameters and initialize testing by asserting reset signals, and when testing is complete interrogating an AHB model for correct operation.

The AHB Memory core (AHBLSRAM) and SPI Flash model are fully functional models. The SPI Flash model is downloaded from a SPI Flash vendor's website.

The Host SPI model initially writes test patterns to SPI Flash memory, which also tests the Host SPI path through the CoreBootStrap core, including reads as reading the status register is performed before writes are done.

The AHB model monitors the AHB bus transactions and checks data against expected patterns, reporting pass/fail information to the tests.

Figure 5 CoreBootStrap User Testbench



6.6 Synthesis in Libero

After setting the design root appropriately for your design, use the following steps to run the Synthesis:

1. Click **Synthesis** in the Libero SoC software. The **Synthesis** window appears, displaying the Synplify® project.
2. Use the Verilog 2001 standard if Verilog is being used.
3. Click **Run** to run the Synthesis.

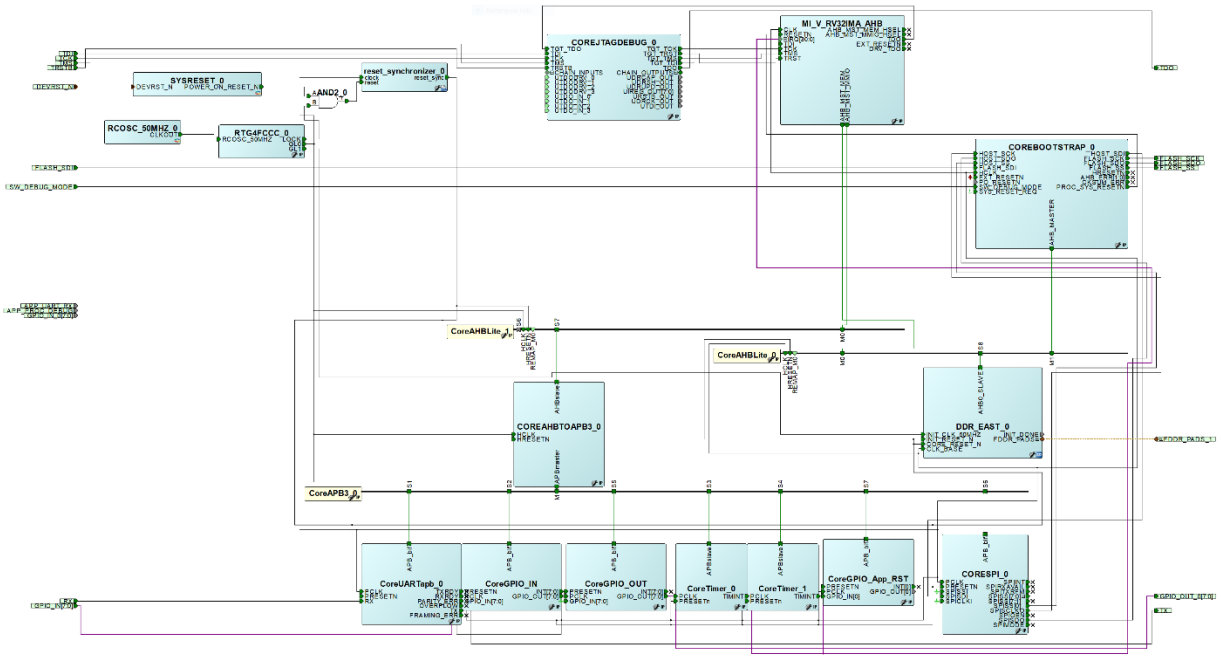
6.7 Place-and-Route in Libero

After setting the design route appropriately for your design, and running Synthesis, click **Layout** in the Libero SoC software to invoke **Designer**. CoreBootStrap requires no special place-and-route settings.

7 System Integration

Figure 6 shows CoreBootStrap placed in a design:

Figure 6 System Integration



a Microchip company
Microsemi 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
email: sales.support@microsemi.com
www.microsemi.com

Microsemi, a wholly owned subsidiary of Microchip Technology Inc. (Nasdaq: MCHP), offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center 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; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services.
Learn more at www.microsemi.com.