Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS and with all faults, and all risk associated with such information is entirely with Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer’s responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice’s product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.
Contents

Acronyms in This Document ................................................................................................................. 4
1. Introduction ........................................................................................................................................... 5
2. Single Port RAM Primitives .................................................................................................................. 5
   2.1. User Primitive SB_SPRAM256KA ................................................................................................. 5
   2.2. SPRAM Port Definitions and User Interface Options ..................................................................... 6
3. Power Save States for SPRAM ............................................................................................................. 8
   3.1. Normal State ................................................................................................................................... 8
   3.2. Standby State ................................................................................................................................. 8
   3.3. Sleep State .................................................................................................................................... 8
   3.4. Power Off State ............................................................................................................................. 8
4. Use Cases for User Primitive SB_SPRAM256KA ................................................................................ 9
   4.1. Instantiating Memories ................................................................................................................... 9
   4.2. Inferring Memories ......................................................................................................................... 9
   4.3. Output Pipeline Registers .............................................................................................................. 9
   4.4. Cascading Memories ...................................................................................................................... 10
      4.4.1. Address Cascading (or Depth Cascading) ................................................................................ 10
      4.4.2. Data Cascading (or Width Cascading) .................................................................................... 11
5. SPRAM Content during Warmboot ...................................................................................................... 11

Technical Support Assistance ................................................................................................................... 12
Revision History ........................................................................................................................................ 13

Figures

Figure 2.1. SB_SPRAM256KA SPRAM Primitive ................................................................................... 5
Figure 4.1. Address/Depth Cascading Example for 32K x 16 SPRAM using Primitive ................................ 10
Figure 4.2. Data/Width Cascading Example for 16K x 32 SPRAM using Primitive .................................... 11

Tables

Table 2.1. SB_SPRAM256KA RAM Port Definitions ............................................................................... 6
### Acronyms in This Document

A list of acronyms used in this document.

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PMU</td>
<td>Power Management Unit</td>
</tr>
<tr>
<td>SPRAM</td>
<td>Single Port RAM</td>
</tr>
<tr>
<td>RAM</td>
<td>Random Access Memory</td>
</tr>
</tbody>
</table>
1. Introduction

The Lattice Semiconductor iCE40™ family of ultra-low power FPGAs features Single Port RAM (SPRAM). This document provides guidance to software engineers on integrating the SPRAM using iCEcube2 software. The iCE40 family has four 256 kb memory blocks available, that is a total of 1024 kb of Single Port memory.

2. Single Port RAM Primitives

The iCE40 devices offer four embedded memory blocks of SPRAM. Each of these blocks can be configured only in 16K x 16 mode. Depending on design requirements, a wrapper is created that instantiates the primitive and connects the ports. These RAM blocks can be cascaded to create larger memories, see the Cascading Memories section.

2.1. User Primitive SB_SPRAM256KA

Each of the four 256 kb blocks of RAM is configured in 16K x 16 Single Port RAM. Figure 2.1 shows the block diagram of the primitive for the 16K x 16 SPRAM block – SB_SPRAM256KA.

![Figure 2.1. SB_SPRAM256KA SPRAM Primitive](image)

ADDRESS [13:0] ➔ DATAOUT [15:0]
DATAIN [15:0]
MASKWREN [3:0]
WREN ➔
CHIPSELECT ➔
CLOCK ➔
STANDBY ➔
SLEEP ➔
POWEROFF ➔

Single Port RAM Primitive
SB_SPRAM256KA
## 2.2. SPRAM Port Definitions and User Interface Options

Table 2.1 shows the port definitions and the user interface options for the SPRAM.

### Table 2.1. SB_SPRAM256KA RAM Port Definitions

<table>
<thead>
<tr>
<th>User Primitive Port Name</th>
<th>Test Primitive Port Name</th>
<th>Primitive Port Width</th>
<th>HW Port Name</th>
<th>HW Port Width</th>
<th>Pin Name</th>
<th>Default Value</th>
<th>Description</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDRESS</td>
<td>—</td>
<td>[13:0]</td>
<td>ADR</td>
<td>[13:0]</td>
<td>Address Input</td>
<td>14b'0000000000000</td>
<td>This Address Input port is used to address the location to be written during the write cycle and read during the read cycle.</td>
<td>—</td>
</tr>
<tr>
<td>DATAIN</td>
<td>—</td>
<td>[15:0]</td>
<td>D</td>
<td>[15:0]</td>
<td>Data Input</td>
<td>16b'0000000000000000</td>
<td>The Data Input bus is used to write the data into the memory location specified by Address input port during the write cycle.</td>
<td>—</td>
</tr>
<tr>
<td>MASKWREN</td>
<td>—</td>
<td>[3:0]</td>
<td>WEM</td>
<td>[15:0]</td>
<td>Maskable Write Enable</td>
<td>4b'1111</td>
<td>It includes the Bit Write feature where selective write to individual I/O can be done using the Maskable Write Enable signals. When the memory is in write cycle, one can write selectively on some I/O.</td>
<td>1, 2</td>
</tr>
<tr>
<td>WREN</td>
<td>—</td>
<td>[0:0]</td>
<td>WE</td>
<td>[0:0]</td>
<td>Write Enable Input</td>
<td>1b'0</td>
<td>When the Write Enable input is Logic High, the memory is in the write cycle. When the Write Enable is Logic Low, the memory is in the read cycle.</td>
<td>—</td>
</tr>
<tr>
<td>CHIPSELECT</td>
<td>—</td>
<td>[0:0]</td>
<td>ME</td>
<td>[0:0]</td>
<td>Memory enable input</td>
<td>1b'0</td>
<td>When the memory enable input is Logic High, the memory is enabled and read/write operations can be performed. When memory enable input is Logic Low, the memory is deactivated.</td>
<td>—</td>
</tr>
<tr>
<td>CLOCK</td>
<td>—</td>
<td>[0:0]</td>
<td>CLK</td>
<td>[0:0]</td>
<td>Clock Input</td>
<td>—</td>
<td>This is the external clock for the memory.</td>
<td>—</td>
</tr>
<tr>
<td>STANDBY</td>
<td>—</td>
<td>[0:0]</td>
<td>LS</td>
<td>[0:0]</td>
<td>Light Sleep Input</td>
<td>1b'0</td>
<td>When this pin is active then memory goes into low leakage mode, there is no change in the output state.</td>
<td>3</td>
</tr>
</tbody>
</table>
### User Primitive Port Name

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Port Name</th>
<th>Primitive Port Name</th>
<th>HW Port Name</th>
<th>HW Port Width</th>
<th>Pin Name</th>
<th>Default Value</th>
<th>Description</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLEEP</td>
<td>—</td>
<td>[0:0] DS</td>
<td>[0:0]</td>
<td></td>
<td>Deep</td>
<td>1b’0</td>
<td>This pin shuts down power to periphery and maintains memory contents. The outputs of the memory are pulled low.</td>
<td>3</td>
</tr>
<tr>
<td>POWEROFF</td>
<td>—</td>
<td>[0:0]</td>
<td>[0:0]</td>
<td></td>
<td>Power Off</td>
<td>1b’1</td>
<td>This pin turns off the power to the memory core. Note that there is no memory data retention when this is driven low.</td>
<td>4</td>
</tr>
<tr>
<td>DATAOUT</td>
<td>—</td>
<td>[15:0] Q</td>
<td>[15:0]</td>
<td></td>
<td>Data Output bus</td>
<td>—</td>
<td>This pin outputs the contents of the memory location addressed by the address input signals.</td>
<td>—</td>
</tr>
</tbody>
</table>

### Notes:

1. **MASKWREN** includes the nibble write masks for the **DATAIN**.
   
   The hardware port, **WEM** allows write mask for individual bits of **DATAIN**.
   
   For external user primitive level, this mask is available for each nibble (4-bits). Thus the **MASKWREN** has to map to **WEM** as follows:
   
   - `MASKWREN(3) => WEM(15)`
   - `MASKWREN(3) => WEM(14)`
   - `MASKWREN(3) => WEM(13)`
   - `MASKWREN(3) => WEM(12)`
   - `MASKWREN(2) => WEM(11)`
   - `MASKWREN(2) => WEM(10)`
   - `MASKWREN(2) => WEM(9)`
   - `MASKWREN(2) => WEM(8)`
   - `MASKWREN(1) => WEM(7)`
   - `MASKWREN(1) => WEM(6)`
   - `MASKWREN(1) => WEM(5)`
   - `MASKWREN(1) => WEM(4)`
   - `MASKWREN(0) => WEM(3)`
   - `MASKWREN(0) => WEM(2)`
   - `MASKWREN(0) => WEM(1)`
   - `MASKWREN(0) => WEM(0)`

2. The default value of each **MASKWREN** is 1. In order to mask a nibble of **DATAIN**, the **MASKWREN** needs to be pulled low (0).
   
   The following shows which **MASKWREN** bits enable the mask for the **DATAIN** nibbles:
   
   - `MASKWREN(3) enables mask for DATAIN(15:12)`
   - `MASKWREN(2) enables mask for DATAIN(11:8)`
   - `MASKWREN(1) enables mask for DATAIN(7:4)`
   - `MASKWREN(0) enables mask for DATAIN(3:0)`

3. **STANDBY**, **SLEEP** signals are mutually exclusive. Refer to the Logic Truth Table (Power Modes) section in the datasheet for valid values for these signals in different states.

4. **POWEROFF** is a signal that controls the built in power switch in each memory block. This signal when driven low (1'b0), shuts down the power to the memory. During the off state, there is no memory data retention.
   
   When **POWEROFF** is driven high (1'b1), the **SPRAM** is powered on.

There are no special attributes needed for **SB_SPRAM256KA** because it has fixed configuration of 16,384 addresses and 16 data width, running in Normal mode. The inputs (**ADDRESS** and **DATAIN**) are always registered. **DATAOUT** has no registers.

**SB_SPRAM256KA** RAM does not support initialization through device configuration.
3. **Power Save States for SPRAM**

The iCE40 provides a capability to place the SPRAM in a different power state, when not in use. There are three user signals that control the power states of the RAM.

### 3.1. Normal State

Normal State is the normal operation of the memory. During this state, all three of the power save signals (STANDBY, SLEEP and SHUTDOWN) are being driven Low. This is also the higher power consumption state of the SPRAM.

### 3.2. Standby State

Standby State is achieved when the STANDBY signal is driven High. When active, the memory goes in a low leakage mode. The state of the outputs does not change when the RAM is placed in Standby State.

It is to be noted that Standby State is referred to as “Light Sleep” state in the RAM datasheet. The name STANDBY has been chosen to match and be consistent with the power states for the Lattice Power Management Unit (PMU).

### 3.3. Sleep State

Sleep State is achieved when the SLEEP signal is driven High. This signal shuts down the power to the periphery of the memory and maintains the memory contents. The outputs in this case are all pulled Low.

Sleep State is referred to as the *Deep Sleep* state in the RAM datasheet. The name SLEEP is chosen to match and be consistent with the power states for the Lattice Power Management Unit (PMU).

### 3.4. Power Off State

Each RAM block has an associated power switch that controls the SD signal of the RAM. Users interface through CIB to the Power Switch and that powers down the memory.

Shut Down or Power Off State is achieved when the POWEROFF signal is driven Low. This signal shuts down the power to the periphery of the memory and the memory core. In this state, there is no data retention of the memory. The outputs in this case are all pulled low.
4. Use Cases for User Primitive SB_SPRAM256KA

This section describes the use cases of the SB_SPRAM256KA RAM blocks while instantiating, inferring, and cascading these blocks.

4.1. Instantiating Memories

SB_SPRAM256KA primitive can be directly instantiated using both Verilog and VHDL at the top level. An example of instantiating SB_SPRAM256KA RAM using Verilog:

```verilog
// spram256 user modules //
SB_SPRAM256KA ramfn_inst1(
    .DATAIN(DATAIN),
    .ADDRESS(ADDRESS),
    .MASKWREN(MASKWREN),
    .WREN(WREN),
    .CHIPSELECT(CHIPSELECT),
    .CLOCK(CLOCK),
    .STANDBY(STANDBY),
    .SLEEP(SLEEP),
    .POWEROFF(POWEROFF),
    .DATAOUT(DATAOUT_A)
)
SB_SPRAM256KA ramfn_inst2(
    .DATAIN(DATAIN),
    .ADDRESS(ADDRESS),
    .MASKWREN(MASKWREN),
    .WREN(WREN),
    .CHIPSELECT(CHIPSELECT),
    .CLOCK(CLOCK),
    .STANDBY(STANDBY),
    .SLEEP(SLEEP),
    .POWEROFF(POWEROFF),
    .DATAOUT(DATAOUT_B)
)
```

4.2. Inferring Memories

The memory also supports memory inferring where a behavioral code for the SPRAM is synthesized in iCEcube2 to create the RAM using the RAM primitives of ICE40 device. In order to use SB_SPRAM256KA RAM blocks, you can use `syn_ramstyle` attribute.

The power save states (Standby, Sleep, and Power Off States) are not available when inferring the RAM. When implementing the inferred RAM using SB_SPRAM256KA primitives, software should tie off the STANDBY, SLEEP and SHUTDOWN ports to “0”. If power save features are desired, use the method of instantiation and connect these ports as per design requirements.

4.3. Output Pipeline Registers

The SB_SPRAM256KA does not include output registers.

When desired, pipeline registers are required to be implemented in the fabric. While inferring the RAM, the software should implement the output pipeline registers in the fabric.
4.4. Cascading Memories

Each SPRAM block is 256 kb, supporting configuration that are 16K x 16. These memories are cascaded to form larger memory based on the user requirements. The memories can be cascaded in two ways:
- Address Cascading or
- Data Cascading.

The following sections provide examples of how each cascading type is achieved, and the connection of the signals required. User can instantiate the RAM primitive and connect them using this as guidance to create larger memory blocks.

Auto cascading is supported while inferring a RAM. Any additional logic required is implemented in the device fabric for creating larger memories.

4.4.1. Address Cascading (or Depth Cascading)

Address/Depth cascading is useful when the memories are required to have the capacity of storing more words while keeping the data width the same. In this case additional user logic is needed to decode the address.

Figure 4.1 shows an example of the depth cascading of a 32K x 16 SPRAM. Additional logic is required that guides the data to the correct memory block using Muxes and Demuxes. The rest of the signals (that are not shown), should be connected to both the memory blocks without any other logic requirements.
4.4.2. Data Cascading (or Width Cascading)

Data/Width cascading is useful when the memories are required to have the capacity of storing longer words while keeping the address depth the same. In this case, minimal user logic is needed, essentially for concatenating words from individual SPRAM blocks.

Figure 4.2 shows an example of the Width cascading of a 16K x 32 SPRAM. The rest of the signals (that are not shown), should be connected to both the memory blocks without any other logic requirements.

Figure 4.2. Data/Width Cascading Example for 16K x 32 SPRAM using Primitive

5. SPRAM Content during Warmboot

During configuration through Warmboot, SPRAM content is not loaded. Thus, previous data on the SPRAM are retained.
Technical Support Assistance
Submit a technical support case through www.latticesemi.com/techsupport.
# Revision History

## Revision 1.2, April 2020

<table>
<thead>
<tr>
<th>Section</th>
<th>Change Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disclaimers</td>
<td>Added this section.</td>
</tr>
<tr>
<td>Acronyms in This Document</td>
<td>Added this section.</td>
</tr>
<tr>
<td>SPRAM Content during Warmboot.</td>
<td>Added this section.</td>
</tr>
<tr>
<td>All</td>
<td>Minor changes in formatting/styles.</td>
</tr>
</tbody>
</table>

## Revision 1.1, August 2017

<table>
<thead>
<tr>
<th>Section</th>
<th>Change Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>• Changed document number from TN1314 to FPGA-TN-02022.</td>
</tr>
<tr>
<td></td>
<td>• Removed copyright page.</td>
</tr>
</tbody>
</table>

## Revision 1.0, June 2016

<table>
<thead>
<tr>
<th>Section</th>
<th>Change Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Initial release.</td>
</tr>
</tbody>
</table>