SMIC Data Sheet

Preliminary

Version:0.2   Release Date: Nov 30, 2013

Semiconductor Manufacturing International Corporation

SMIC Data Sheet

Preliminary

Notice

DISCLAIMER

STATEMENT OF USE AND CONFIDENTIALITY

Semiconductor Manufacturing International Corporation
No. 18 Zhangjiang Road
Pudong New Area
Shanghai 201203
The People's Republic of China
Email: Design_Service@smics.com
URL: www.smics.com

Chapter 1 Installing and Using the Compiler

1.1 Introduction

This user guide provides the information of SMIC 130nm Single-Port SRAM compiler about how to generate embedded SRAM macros and corresponding Design Kits.

1.2 Memory Compiler Features

- □ High density
- □ Optimized power distribution scheme (Over the cell power routing)
- □ Bits write option
- □ Low active power and low standby leakage power
- □ Timing and power models for advanced design tools

1.3 Compiler Installation

This section contains System Environment Requirements and Installation Instructions for the SMIC 130nm Single-Port SRAM Compiler.

1.3.1 System Environment Requirements

Requires Java Version 1.6.0_12(JDK 1.6.0_12) or higher.
To verify your Java version using the following command:

```
% java –version
```

1.3.2 Installation Instructions

- The SMIC 130nm Single-Port Memory Compiler installation package: S013LLLPSP.tar.gz, includes the following files:

```
1.    S013LLLPSP.jar          # main memory compiler application
2.    S013LLLPSP.csh          # setup file for C shell
3.    lib/                    # necessary library folder
4.    reffile/                #necessary reference file
5.    S013LLLPSP_ug.pdf       #user guide
6.    S013LLLPSP.notes        #release note
```

- Installation Procedure:

1. Copy the installation package into a stand-alone directory.
2. Uncompress and un-tar the application package:

```
% gunzip < S013LLLPSP.tar.gz | tar xvf -
```

or

```
% gtar xzvf S013LLLPSP.tar.gz
```

3. Verify that the files in the install directory are executable.

```
% chmod -R 755 /your_directory/
```

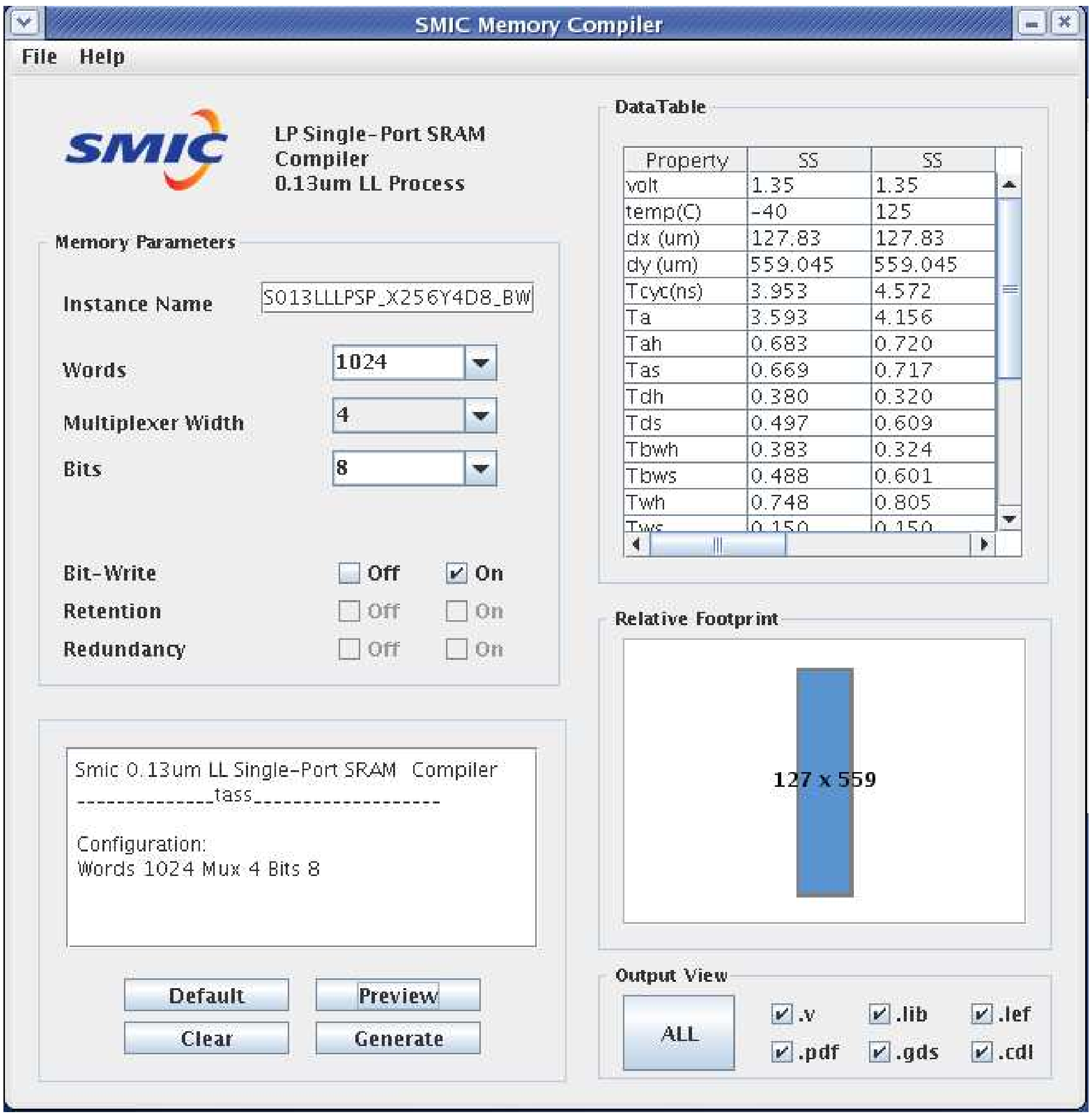
1.4 Launching the Compiler from GUI

Invoke the SMIC Memory Generator GUI in /your directory/ using the following command with no arguments:

```
% ./S013LLLPSP.csh
```

1.5 Compiler Window

Example of the SMIC 130nm Single-Port Memory Compiler GUI is shown in Figure 1-1.



SMIC Memory Compiler

File Help

LP Single-Port SRAM Compiler 0.13um LL Process

Memory Parameters

Instance Name	S013LLLPSP_X256Y4D8_BW	
Words	1024	
Multiplexer Width	4	
Bits	8	
Bit-Write	Off	[✓ On]
Retention	Off	On
Redundancy	Off	On

```
Smic 0.13um LL Single-Port SRAM  Compiler
--------------tass------------------

Configuration:
Words 1024 Mux 4 Bits 8
```

DataTable

Property	SS	SS
volt	1.35	1.35
temp(C)	-40	125
dx (um)	127.83	127.83
dy (um)	559.045	559.045
Tcyc(ns)	3.953	4.572
Ta	3.593	4.156
Tah	0.683	0.720
Tas	0.669	0.717
Tdh	0.380	0.320
Tds	0.497	0.609
Tbwh	0.383	0.324
Tbws	0.488	0.601
Twh	0.748	0.805
Tws	0.150	0.150

Relative Footprint

127 x 559

Output View

ALL [✓ .v] [✓ .lib] [✓ .lef] [✓ .pdf] [✓ .gds] [✓ .cdl]

Figure 1-1 SMIC Memory Compiler GUI

1.5.1 Memory Parameters Panel

The Memory Parameters panel contains input field and check boxes. You can change values of *Word*, *Multiplexer* and *Bits* by selecting the value you want from the pull-down submenu corresponding to each parameter or typing the value in the input field. Also you could select *off* or *on* for *Bit-Write* in check boxes.

When you finish filling the parameter values, you could view the update *DataTable* panel, *Message* panel and *Relative Footprint* panel by click *Preview* button.

For Example, you could generate a view for a specific instance with 1024 words, 8 bits and 4 Multiplexer Width. Select 1024 from *Words* pull-down submenu, 4 from *Multiplexer Width* pull-down submenu and 8 from *Bits* pull-down submenu. Click

Preview button. Then *DataTable* panel, *Message* panel and *Relative Footprint* panel will be updated.

1.5.2 Output View Panel

You could generate multiple views at a time. To generate a single view or multiple views, select the view you want from the *Output View* checkboxes. Click the *Generate* button, then *save as*… submenu will pop up, enter path or folder name, click OK. The view will be generated and be saved at the path you entered just now.

1.5.3 Relative Footprint Panel

The *Relative Footprint* panel shows the aspect ratio of the corresponding specific instance. When you change the *Memory parameters* and click *Preview* button, the *Relative Footprint* will be updated.

1.5.4 Data Table Panel

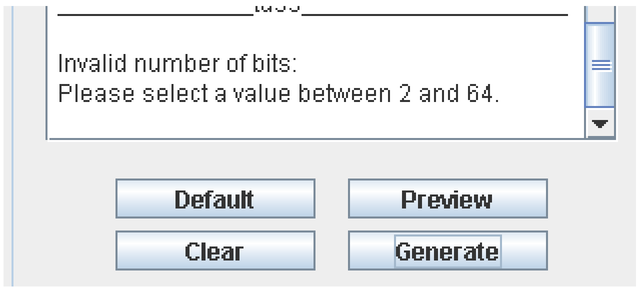
The *Data Table* shows corners, layout size, timing, current, and power information for the corresponding specific instance.

When you change the *Memory parameters* and click *Preview* button, the *Data Table* will be updated.

1.5.5 Message Panel

The *Message* panel shows messages when you generate an instance. The message includes the configuration information of the instance be generating, the associated view type information be generating and successful generation information.

The *Message* panel also report problem when you enter the values out of the pre-determined range. For Example, when enter 322 in the *bits* input field, it will report the following message (see Figure 1-2).



Invalid number of bits:
Please select a value between 2 and 64.

Figure 1-2 Message Panel

1.5.6 File Menu

To exit the Compiler GUI, select *File* menu, then select *Exit* submenu.

1.6 Views and Output Files

Table 1-1 lists the kinds of views you could generate and the corresponding output files.

Table 1-1 Views and Output Files

View	Output Files
LEF footprint	<instance name>.lef
Synopsys model	<instance name>.lib
PDF datasheet	<instance name>.pdf
Verilog model	<instance name>.v
GDSII	<instance name>.gds
LVS Netlist	<instance name>.cdl

1.7 Generating Views

You can generate single or multiple views from GUI and command line.

1.7.1 Generating Views from GUI

You can generate multiple views from GUI. To generate a new instance, change the *Memory parameters*, click *Preview* button. Select a view from *Output View* checkbox, click *Generate* button. The view will be generated.

1.7.2 Generating Views from Command Line

Generate single or multiple output views from the command line by specifying additional arguments. Enter " java –jar S013LLLPSP.jar -help" for usage options:

```
% java –jar S013LLLPSP.jar [options...]
```

- -bitwrite : toggle bitwrite
- -nov :generate typical voltage is 1.2v lib and pdf datasheet
- -help : display usage options
- -bits VAL : number of bits
- -words VAL : number of words
- -mux VAL : column multiplexer value
- -lef : generates .lef footprint
- -lib : generates Synopsys .lib model
- -pdf : generates pdf datasheet
- -v : generates verilog model
- -gds : generate GDSII
- -cdl : generate LVS netlist

- -savepath : define saved file path.
- -instname : define instance name.

-words , -mux , -bits and at least one output view must be specified

Ex. % java –jar S013LLLPSP.jar -words 4096 -mux 8 -bits 32 -v -lef -lib –pdf

Chapter 2 Synchronous SRAM Compiler

2.1 Synchronous Single-Port SRAM Structure and Timing Specifications

2.1.1 Single-Port SRAM Description

2.1.1.1 Basic Functionality

The SRAM access is synchronous and is triggered by the rising edge of the clock. The write enable (WEN) and bit-write enable (BWEN[0:n]), chip enable (CEN), address (A[0:i]), and data in (D[0:n]) signals are latched on the rising edge of the clock.

- If bit-write is off:

When CEN is low and WEN is high the memory will be in read operation. Data is read from the location specified by the address A[0:i], and is output on the output port Q[0:n].

When CEN and WEN are both low the memory will be in write operation. The word on the data port D[0:n] will be written into the location specified by the address A[0:i] and the data will appear on the output port Q[0:n].

When CEN is high the memory is in standby mode. Meanwhile, the data stored in memory is retained but cannot be read or written.

- If bit-write is on

When CEN is low and WEN is high, the memory will be in read operation. Data is read from the location specified by the address A[0:i], and is output on the output port Q[0:n].

When CEN is low, WEN is low and BWEN[j] is high, the memory will be in read operation. Data D[j] is read from the location specified by the address A[0:i], and is output on the output port Q[j].

When CEN is low, WEN is low and BWEN[j] is low the memory will be in write operation, the corresponding data on the data port D[j] will be written into the location specified by the address A[0:i] and the data will appear on the corresponding output port Q[j].

When CEN is high the memory is in standby mode. Meanwhile, the data stored in memory is retained but cannot be read or written.

For instance, a memory with 16 bits. When CEN is low meanwhile the WEN is low, the bit-write enable BWEN [0:7] is low and BWEN[8:15] is high, the memory will be in write operation. The word on D[0:7] will be written into the location specified by the address A[0:i] and the data will appear on the output port Q[0:7]. D[8:15] could not be written into the memory and could not be output on the output port. The output port Q[8:15] is the result of a read operation.

2.1.1.2 Test and Repair Functionality

- **Repair/Redundancy**

The Repair Feature (also called Redundancy) supports one row redundancy in the memory array block. The redundant row could replace any other row.
Figure2-1 shows the structure of a Single-Port SRAM with one row redundancy at bottom of a memory array.

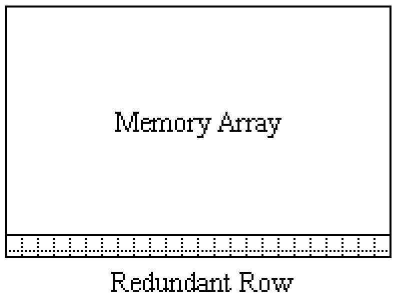


Figure 2-1 The structure of a Single-Port SRAM with one row redundancy

When a defective row is found, the row is replaced by the redundant row. The memory with redundancy has added signal pin RDE. When the location specified by the address A[0:i] matches the defect address pattern RDE will be enabled and the redundant row will be accessed.

2.1.2 Single-Port SRAM Pins

Figure 2-2 shows the basic Single-Port SRAM Pins.

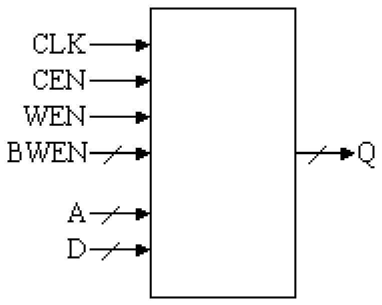


Figure 2-2 Single-Port SRAM Basic Pins

Table2-1 shows the basic Single-Port SRAM Pins Definition.

Table2-1 the basic Pins Definition

PIN	**DIRECTION**	**DEFINITION**
A [0:i]	Input	Address Inputs
D [0:n]	Input	Data Inputs
WEN	Input	Write Enable
BWEN [0:n]	Input	Bit-Write Enable
CEN	Input	Chip Enable
CLK	Input	Clock Input
Q [0:n]	Output	Data Outputs

2.1.3 Single-Port SRAM Logic Table

This section shows logic tables for Single-Port SRAM basic functions and repair functions.

- 130nm Single-Port SRAM basic functions are described in Table 2-2 and Table 2-3

Table 2-2 130nm Single-Port SRAM basic functions (Bit-Write is off)

CEN	WEN	Operation	Output	Function
Low	High	Read	Stored data	Data D[0:n] is read from the location specified by the address A[0:i] and is output on the output port Q[0:n]
Low	Low	Write	Data In	The word on the data port D[0:n] is written into the memory location specified by the address A[0:i] and is output on the output port Q[0:n]
High	X	Standby	Last Data	The data stored in memory is retained but cannot be read or written

Table 2-3 130nm Single-Port SRAM basic functions (Bit-Write is on)

CEN	**WEN**	**BWEN [i]**	**Operation**	**Output**	**Function**
Low	High	X	Read	Stored data	Data D[0:n] is read from the location specified by the address A[0:i] and is output on the output port Q[0:n]
Low	Low	High	Read	Stored data	Data D[i] is read from the location specified by the address A[0:i] and is output on the output port Q[i]

Low	Low	Low	Write	Data In	The corresponding data on the data port D[i] will be written into the memory location specified by the address A[0:i] and the data will appear on the corresponding output port Q[i]
High	X	X	Standby	Last Data	The data stored in memory is retained but cannot be read or written

- 130nm Single-Port SRAM repair functions are described in Table 2-4

Table 2-4 130nm Single-Port SRAM repair functions

RDE	**Operation**	**Function**
High	Redundant row enabled	When the location specified by the address A[0:i] matches the defective address pattern RDE will be enabled and the redundant row will be accessed.
Low	Redundant row disabled	Normal function

2.1.4 130nm Single-Port SRAM Parameters

The 130nm Single-Port SRAM standard input and block parameters are listed in Table 2-5. You could refer to Compiler GUI for pre-determined input ranges. If you enter an invalid value and *Preview* the window, a problem message and the pre-determined range of the value will be displayed in the *Message* panel.

Table 2-5 130nm Single-Port SRAM Parameters

Parameters	Ranges	
Words Number	Ymux = 4	32~1280, increment = 8*Ymux
	Ymux = 8	64~2560, increment = 8*Ymux
Bits Number	Ymux = 4	2~64, increment = 1
	Ymux = 8	2~64, increment = 1
MUX	8, 4	
Bit-Write	on, off; default off	
Top chip metal layers support	m4 to top metal layer supported by design process	
Power type	m4 power strap	

2.1.5 130nm Single-Port SRAM Block Diagram

The instance block diagram of 130nm Single-Port SRAM is shown in Figure 2-3

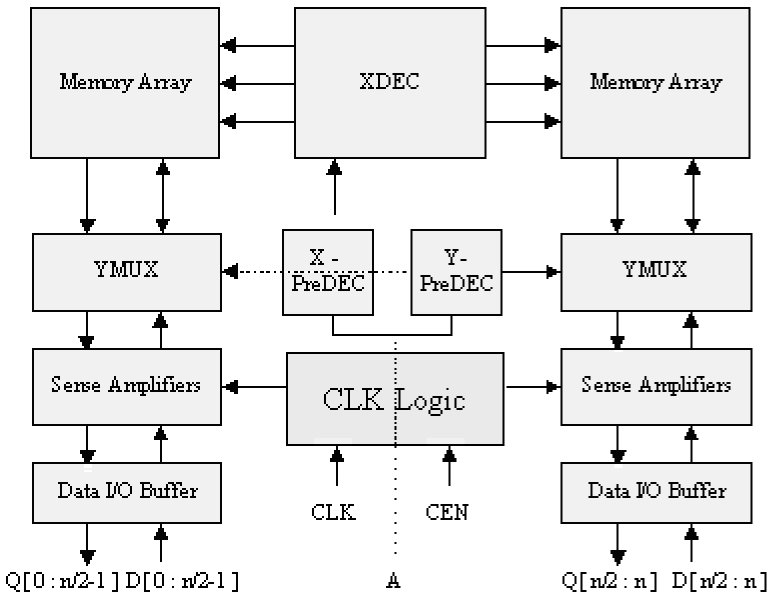


Figure 2-3 130nm Single-Port SRAM Block Diagram

2.1.7 130nm Single-Port SRAM Timing Description

This section describes the timing diagrams, timing parameters and power parameters of 130nm Single-Port SRAM.

2.1.7.1 130nm Single-Port SRAM Timing Diagrams

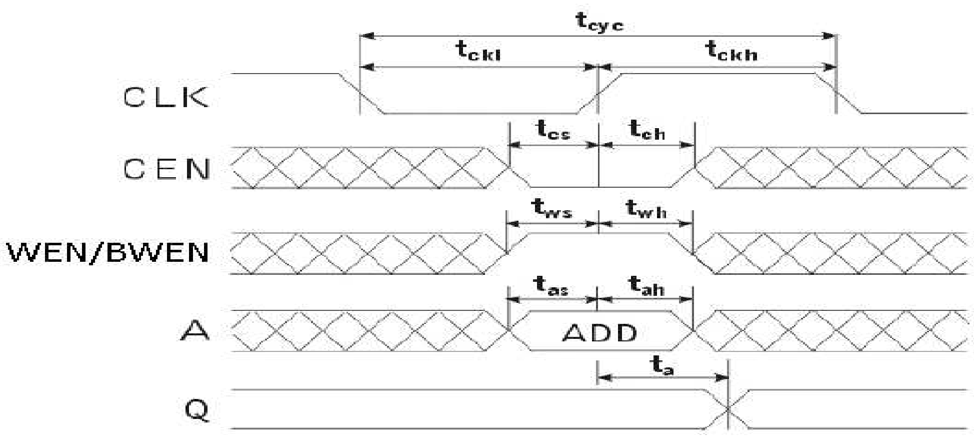


Figure 2-8 130nm Single-Port SRAM Read-Cycle Timing

[WEN is a signal pin. (bit-write is off)BWEN is a bus. (bit-write is on)]

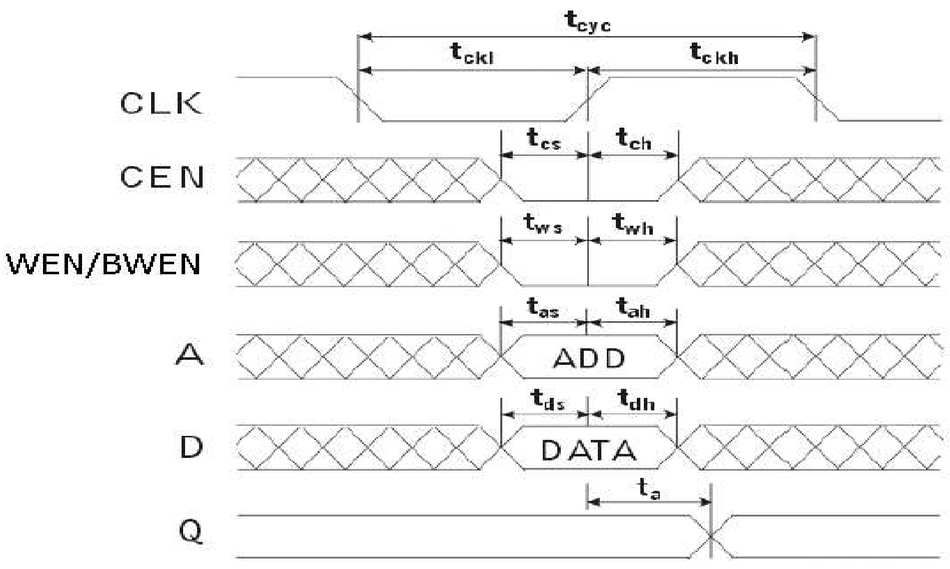


Figure 2-9 130nm Single-Port SRAM Write-Cycle Timing,
[WEN is a signal pin. (bit-write is off), BWEN is a bus. (bit-write is on)]

2.1.7.2 130nm Single-Port SRAM Timing Parameters

The timing parameters that show up in GUI *Data Table* are listed in Table 2-6.

Table 2-6 130nm Single-Port SRAM Timing Parameters

Parameter (ns)	**Description**
Tcyc	Cycle Time
Ta	Access Time
Tas	Address Setup
Tah	Address Hold
Tds	Data Setup
Tdh	Data Hold
Tcs	Chip Enable Setup
Tch	Chip Enable Hold
Tws	Write Enable Setup
Twh	Write Enable Hold
Tckh	Clock High
Tckl	Clock Low
Tckr	Clock Rise Skew
Tbws	Bit Write Enable Setup
Tbwh	Bit Write Enable Hold

2.1.7.3 130nm Single-Port SRAM Power Parameters

The timing parameters that show up in GUI *Data Table* are listed in Table 2-7.

Table 2-7 130nm Single-Port SRAM Power Parameters

Parameter (uA/Mhz)	**Description**
icc	AC Current

iccr	Read AC Current
iccw	Write AC Current
isb	Standby Power (mW)

2.2 130nm Single-Port SRAM Power Structure

2.2.1 Current Calculations

2.2.1.1 Average Current

The average current, Iavg, which is reported in the datasheet and GUI timing table can be calculated by following formular:

I_avg=I_avg_read*M_read_percentage+I_avg_write*N_write_percentage+
(C_pin*V*F)*S_data_switch_percentage

Where,

- ☐ I_avg_read= average read current (mA)
- ☐ I_avg_write= average write current (mA)
- ☐ M_read_percentage= percentage of read operation (%)
- ☐ N_write_percentage= percentage of write operation (%)
- ☐ S_data_switch_percentage= percentage of input and output pins switch (%)
- ☐ C_pin = average capacitance of output port (pF)
- ☐ V= power supply voltage (V)
- ☐ F= operation frequency (Hz)

The Iavg in our datasheet is calculated by the assumption of 100% address switch, M=N=S=50%.
User can recalculate the I_avg base on the special value of M, N and S in your design.

2.2.1.2 Peak current

The peak current, Ipeak is calculated by the SPICE simulator during the Read/Write operation. The Ipeak is the simulated value and the duration is short.

2.2.1.3 Standby current

The standby current, Istandby, is measured from following operation condition:
CEN is disable, all of the input pins are stable except CLK.

2.2.2 Power Distribution Methodology

User's full chip level power mesh which connect to SRAM macro must meet the spec of EM(electro-migration) rules and the limitation of average and peak IR drop. The

voltage supplied to the SRAM macro must be the same value as the characterized voltage to make sure the SRAM speed accuracy.
User can calculate the minimum power bus width connect to SRAM as following formulars:
(This formular is not include the other elements which could consume power through the same power wire. User need to take it into account for minimum power bus calculation)

- W = max (WEM, WIR_PEAK, WIR_AVERAGE)
- W_EM = I_avg/D (um)
- W_IR_PEAK = I_peak*(R_square* L_effective_power)/⊿V_IR_PEAK (um)
- W_IR_AVERAGE = I_avg*(Rsquare* L_effective_power)/⊿V_IR_AVERAGE (um)

Where,

- W = Final minimum power bus width
- W_EM = Power bus width based on EM rules limitation (um)
- W_IR_AVERAGE= Power bus width based on average current IR drop voltage limitation (um)
- W_IR_PEAK = Power bus width based on peak current IR drop voltage limitation (um)
- I_avg = Average current of SRAM macro (mA)
- I_peak = Peak current of SRAM macro (mA)
- D = The value of metal current density rules (uA/um)
- R_square = Square resistance of metal (Ohm/square)
- L_ffective_power = The effective power bus connection length from IO power PAD to the SRAM macro (um)
- ⊿V_IR_PEAK = The spec of peak current IR drop of the chip level power mesh (mV)
- ⊿V_IR_AVERAGE = The spec of average current IR drop of the chip level power mesh (mV)

2.2.3 Noise Limits

The characterized clock noise limit,vn_ck, is the maximum CLK voltage allowable for the indicated pulse width without causing a spurious memory cycle or other memory failure. The standard pulse width, pwn_ck, used in characterizing this limit is 10ns.
The power and ground noise limits,vn_pwr and vn_gnd respectively, are the maximum supply or ground voltage transition allowable without causing a memory failure. Power and ground noise limits are assured at 10% of the characterized voltage.

2.3 130nm Single-Port SRAM Physical Characteristics

This section indicates physical characteristics for 130nm Single-Port SRAM.

2.3.1 TOP Metal Layer

The Compiler supports different top metal layer designs. User has the flexibility to choose M5, M6, M7, M8 or M9 as top metal base on user’s design spec.

M5 and above layers can be used as routing resource over SRAM layout. M4 and below layers is used for SRAM layout design, therefore, are blocked for routing.

2.3.2 Pin Connections

All of the signal pins of the SRAM macro are located on the bottom of the block. And the signal pins can be accessed by the router from M1 to M3 routing layers.

2.3.3 Characterization Environments

The generator is characterized at FF, TT, and SS corners. SMIC recommends user to perform setup, hold and critical path timing analyses at all applicable corners.

Chapter 3 Compiler Views

3.1 Overview

This chapter lists the EDA tools used for verification and the tools supported by the Compiler.

3.2 Tool Verification

This section introduces the views generated by the Compiler and the tools used for verification.

Table3-1 Views and corresponding verification tools

Views	**Tool**
Verilog(.v)	VCS (Synopsys)
Liberty model(.lib)	Design Compiler (Synopsys)
LEF file(.lef)	SOC Encounter (Cadence)
GDSII and Netlist	Calibre (Mentor)

3.3 Using the Views

3.3.1 Using Verilog Model

When finish generating the Verilog model, you could simulate the file using VCS.

Simulation:

```
    %   vcs   <TestBench>.v
    %   simv > verilog.log
```

The simulation output will be put into a file verilog.log.

Also you could check syntax of the Verilog model using VCS.
Check syntax:

```
   %   vcs <instance_name>.v
   %   simv
```

where <instance_name>.v is the verilog model file.

3.3.2 Using LEF file

LEF file provides an abstract view for the memory. It contains the information required by Cadence placement and routing tools. This abstract view defines the size of the instance, pin names, pin locations and layers information.

LEF view only contains MACRO definition. You should have a technology file which contains layer names, layer properties and via definitions. The technology file should be loaded into the router before reading the LEF file.

You could also load the LEF file into Milkyway.
Start Milkyway. You could type in *read_lef* in the command window or select menu *Cell Library -> Lef In…*, a submenu will popup, fill in the .lef file name, then click OK to create FRAM view and CEL view.

3.3.3 Using Liberty model

The Compiler could generate TT/FF/SS .lib files. The SS Synopsys model is generated with maximum delays, and the FF model is generated with minimum delays.

The Liberty model contains delay model, environment conditions and lookup table. Delay model provides information for lookup table delay calculation. Environment conditions provide operating conditions, default attributes and k-factors for circuit timing evaluation. Lookup table describes pin capacitance, timing and power information.

You could generate SDF from .lib using the following steps:
Start Synopsys Design Compiler:

```
% dc_shell
```

You will in the Synopsys Design Compiler command window, type in the following commands:

```
%   read_lib <instance_name>.lib
%   write_lib <userdefine>
%   link_library=<userdefine>.db
%   target_library=<userdefine>.db
%   read –f verilog <instance_name>.v
%   write_timing –context verilog –f sdf –v2.1 –0 <out_file_name>
```

Where <instance_name>.lib is the Liberty file, <userdefine> is the name of your library, <instance_name>.v is the verilog file, <out_file_name> is the name of out put file.

3.3.4 Using GDSII

The GDSII file contains layouts of the instances. It is GDSII format and contains the mask data. It could also be used to do LVS check.

You could also load the GDSII file into Milkyway.
Start Milkyway. You should already read the .lef file into the Milkyway before this. You could type in auStreamIn in the command window or select menu Cell Library -> Stream in… to read in the GDSII file. This will overwrite the CEL view with the GDSII layout.

3.3.5 Using Netlist

After finish generating the Netlist, you could use it for LVS check or function simulation.

Usually, you may use Calibre(Mentor Graphics) to do LVS check.
The netlist could be added to the chip level netlist, then you could do chip level LVS check.