# iCE40™ LP/HX/LM Family Handbook

HB1011 Version 01.2, November 2013

# iCE40 LP/HX/LM Family Handbook Table of Contents

**October 2013**

## Section I. iCE40 LP/HX Family Data Sheet

# Section I. iCE40 LP/HX Family Data Sheet

DS1040 Version 02.7, October 2013

# iCE40 LP/HX Family Data Sheet Introduction

October 2013 Data Sheet DS1040

## Features

- **Flexible Logic Architecture**
  - Five devices with 384 to 7,680 LUT4s and 10 to 206 I/Os
- **Ultra Low Power Devices**
  - Advanced 40 nm low power process
  - As low as 21 µW standby power
  - Programmable low swing differential I/Os
- **Embedded and Distributed Memory**
  - Up to 128 Kbits sysMEM™ Embedded Block RAM
- **Pre-Engineered Source Synchronous I/O**
  - DDR registers in I/O cells
- **High Performance, Flexible I/O Buffer**
  - Programmable sysIO™ buffer supports wide range of interfaces:
    - LVCMOS 3.3/2.5/1.8
    - LVDS25E, subLVDS
    - Schmitt trigger inputs, to 200 mV typical hysteresis
  - Programmable pull-up mode
- **Flexible On-Chip Clocking**
  - Eight low-skew global clock resources
  - Up to two analog PLLs per device
- **Flexible Device Configuration**
  - SRAM is configured through:
    - Standard SPI Interface
    - Internal Nonvolatile Configuration Memory (NVCM)
- **Broad Range of Package Options**
  - WLCSP, QFN, VQFP, TQFP, ucBGA, caBGA, and csBGA package options
  - Small footprint package options
    - As small as 1.40x1.48mm
  - Advanced halogen-free packaging

***Table 1-1. iCE40 Family Selection Guide***

| Part Number | | LP384 | LP640 | LP1K | LP4K | LP8K | HX1K | HX4K | HX8K |
|---|---|---|---|---|---|---|---|---|---|
| Logic Cells (LUT + Flip-Flop) | | 384 | 640 | 1,280 | 3,520 | 7,680 | 1,280 | 3,520 | 7,680 |
| RAM4K Memory Blocks | | 0 | 8 | 16 | 20 | 32 | 16 | 20 | 32 |
| RAM4K RAM bits | | 0 | 32K | 64K | 80K | 128K | 64K | 80K | 128K |
| Phase-Locked Loops (PLLs) | | 0 | 0 | 1[1] | 2[2] | 2[2] | 1[1] | 2 | 2 |
| Maximum Programmable I/O Pins | | 63 | 25 | 95 | 167 | 178 | 95 | 95 | 206 |
| Maximum Differential Input Pairs | | 8 | 3 | 12 | 20 | 23 | 11 | 12 | 26 |
| High Current LED Drivers | | 0 | 3 | 3 | 0 | 0 | 0 | 0 | 0 |
| **Package** | **Code** | **Programmable I/O: Max Inputs (LVDS25)** | | | | | | | |
| 16 WLCSP (1.40 x 1.48mm, 0.35mm) | SWG16 | | 10(0) | 10(0) | | | | | |
| 32 QFN (5 x 5mm, 0.5mm) | SG32 | 21(3) | | | | | | | |
| 36 ucBGA (2.5 x 2.5mm, 0.4mm) | CM36 | 25(3) | | 25(3)[1] | | | | | |
| 49 ucBGA (3 x 3mm, 0.4mm) | CM49 | 37(6) | | 35(5)[1] | | | | | |
| 81 ucBGA (4 x 4mm, 0.4mm) | CM81 | 55(3) | | 63(8) | 63(9)[2] | 63(9)[2] | | | |
| 81 csBGA (5 x 5mm, 0.5mm) | CB81 | | | 62(9)[1] | | | | | |

*Table 1-1. iCE40 Family Selection Guide (continued)*

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 84 QFN<br>(7 x 7mm, 0.5mm) | QN84 | | | 67(7)[1] | | | | | |
| 100 VQFP<br>(14 x 14mm, 0.5mm) | VQ100 | | | | | | 72(9)[1] | | |
| 121 ucBGA<br>(5 x 5mm, 0.4mm) | CM121 | | | 95(12) | 93(13) | 93(13) | | | |
| 121 csBGA<br>(6 x 6mm, 0.5mm) | CB121 | | | 92(12) | | | | | |
| 132 csBGA<br>(8 x 8mm, 0.5mm) | CB132 | | | | | | 95(11) | 95(12) | 95(12) |
| 144 TQFP<br>(20 x 20mm, 0.5mm) | TQ144 | | | | | | 96(12) | 107(14) | |
| 225 ucBGA<br>(7 x 7mm, 0.4mm) | CM225 | | | | 178(23) | 178(23) | | | 178(23) |
| 256-ball caBGA<br>(14 x 14mm, 0.8mm) | CT256 | | | | | | | | 206(26) |

1. No PLL available on the 16 WLCSP, 36 ucBGA, 81 csBGA, 84 QFN and 100 VQFP packages.
2. Only one PLL available on the 81 ucBGA package.
3. High Current I/Os only available on the 16 WLCSP package.

## Introduction

The iCE40 family of ultra-low power, non-volatile FPGAs has five devices with densities ranging from 384 to 7680 Look-Up Tables (LUTs). In addition to LUT-based, low-cost programmable logic, these devices feature Embedded Block RAM (EBR), Non-volatile Configuration Memory (NVCM) and Phase Locked Loops (PLLs). These features allow the devices to be used in low-cost, high-volume consumer and system applications. Select packages offer High-Current drivers that are ideal to drive three white LEDs, or one RGB LED.

The iCE40 devices are fabricated on a 40 nm CMOS low power process. The device architecture has several features such as programmable low-swing differential I/Os and the ability to turn off on-chip PLLs dynamically. These features help manage static and dynamic power consumption, resulting in low static power for all members of the family. The iCE40 devices are available in two versions – ultra low power (LP) and high performance (HX) devices.

The iCE40 FPGAs are available in a broad range of advanced halogen-free packages ranging from the space saving 1.40x1.48mm WLCSP to the PCB-friendly 20x20 mm TQFP. Table 1-1 shows the LUT densities, package and I/O options, along with other key parameters.

The iCE40 devices offer enhanced I/O features such as pull-up resistors. Pull-up features are controllable on a “per-pin” basis.

The iCE40 devices also provide flexible, reliable and secure configuration from on-chip NVCM. These devices can also configure themselves from external SPI Flash or be configured by an external master such as a CPU.

Lattice provides a variety of design tools that allow complex designs to be efficiently implemented using the iCE40 family of devices. Popular logic synthesis tools provide synthesis library support for iCE40. Lattice design tools use the synthesis tool output along with the user-specified preferences and constraints to place and route the design in the iCE40 device. These tools extract the timing from the routing and back-annotate it into the design for timing verification.

Lattice provides many pre-engineered IP (Intellectual Property) modules, including a number of reference designs, licensed free of charge, optimized for the iCE40 FPGA family. By using these configurable soft core IP cores as standardized blocks, users are free to concentrate on the unique aspects of their design, increasing their productivity.

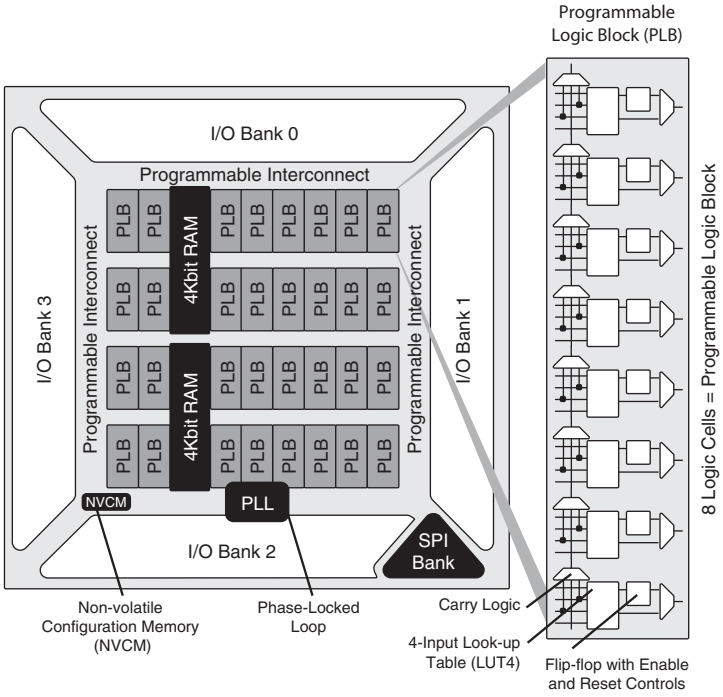
# iCE40 LP/HX Family Data Sheet Architecture

**October 2013** **Data Sheet DS1040**

## Architecture Overview

The iCE40 family architecture contains an array of Programmable Logic Blocks (PLB), sysCLOCK™ PLLs, Non-volatile Programmable Configuration Memory (NVCM) and blocks of sysMEM™ Embedded Block RAM (EBR) surrounded by Programmable I/O (PIO). Figure 2-1 shows the block diagram of the iCE40LP/HX1K device.

***Figure 2-1. iCE40LP/HX1K Device, Top View***




The logic blocks, Programmable Logic Blocks (PLB) and sysMEM EBR blocks, are arranged in a two-dimensional grid with rows and columns. Each column has either logic blocks or EBR blocks. The PIO cells are located at the periphery of the device, arranged in banks. The PLB contains the building blocks for logic, arithmetic, and register functions. The PIOs utilize a flexible I/O buffer referred to as a sysIO buffer that supports operation with a variety of interface standards. The blocks are connected with many vertical and horizontal routing channel resources. The place and route software tool automatically allocates these routing resources.

In the iCE40 family, there are up to four independent sysIO banks. Note on some packages $V_{CCIO}$ banks are tied together. There are different types of I/O buffers on the different banks. Refer to the details in later sections of this document. The sysMEM EBRs are large 4 Kbit, dedicated fast memory blocks. These blocks can be configured as RAM, ROM or FIFO.

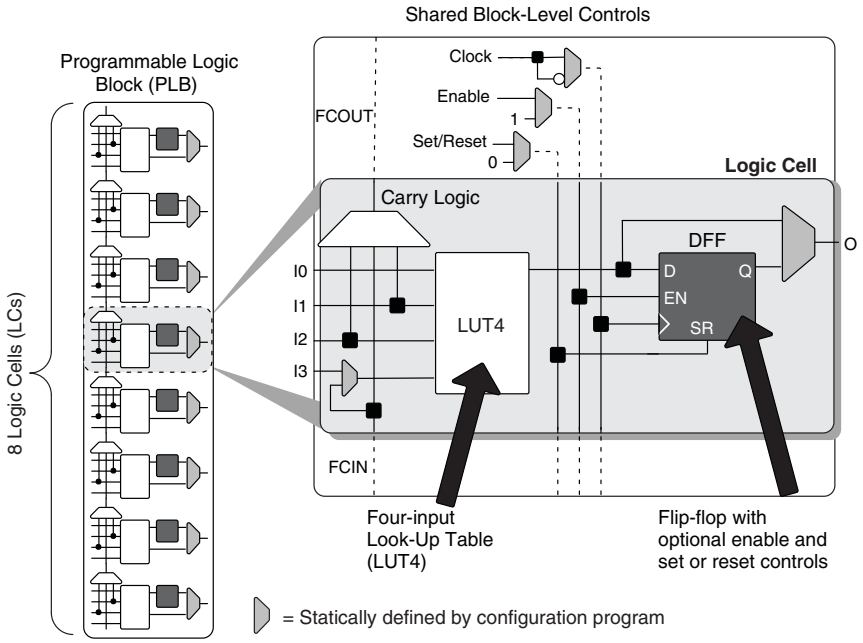
The iCE40 architecture also provides up to two sysCLOCK Phase Locked Loop (PLL) blocks. The PLLs have multiply, divide, and phase shifting capabilities that are used to manage the frequency and phase relationships of the clocks.

Every device in the family has a SPI port that supports programming and configuration of the device. The iCE40 includes on-chip, Nonvolatile Configuration Memory (NVCM).

## PLB Blocks

The core of the iCE40 device consists of Programmable Logic Blocks (PLB) which can be programmed to perform logic and arithmetic functions. Each PLB consists of eight interconnected Logic Cells (LC) as shown in Figure 2-2. Each LC contains one LUT and one register.

***Figure 2-2. PLB Block Diagram***




### Logic Cells

Each Logic Cell includes three primary logic elements shown in Figure 2-2.

- A four-input Look-Up Table (LUT4) builds any combinational logic function, of any complexity, requiring up to four inputs. Similarly, the LUT4 element behaves as a 16x1 Read-Only Memory (ROM). Combine and cascade multiple LUT4s to create wider logic functions.
- A 'D'-style Flip-Flop (DFF), with an optional clock-enable and reset control input, builds sequential logic functions. Each DFF also connects to a global reset signal that is automatically asserted immediately following device configuration.
- Carry Logic boosts the logic efficiency and performance of arithmetic functions, including adders, subtracters, comparators, binary counters and some wide, cascaded logic functions.

***Table 2-1. Logic Cell Signal Descriptions***

| Function | Type | Signal Names | Description |
|---|---|---|---|
| Input | Data signal | I0, I1, I2, I3 | Inputs to LUT4 |
| Input | Control signal | Enable | Clock enable shared by all LCs in the PLB |
| Input | Control signal | Set/Reset[1] | Asynchronous or synchronous local set/reset shared by all LCs in the PLB. |
| Input | Control signal | Clock | Clock one of the eight Global Buffers, or from the general-purpose interconnects fabric shared by all LCs in the PLB |
| Input | Inter-PLB signal | FCIN | Fast carry in |
| Output | Data signals | O | LUT4 or registered output |
| Output | Inter-PFU signal | FCOUT | Fast carry out |

1. If Set/Reset is not used, then the flip-flop is never set/reset, except when cleared immediately after configuration.

## Routing

There are many resources provided in the iCE40 devices to route signals individually with related control signals. The routing resources consist of switching circuitry, buffers and metal interconnect (routing) segments.

The inter-PLB connections are made with three different types of routing resources: Adjacent (spans two PLBs), x4 (spans five PLBs) and x12 (spans thirteen PLBs). The Adjacent, x4 and x12 connections provide fast and efficient connections in the diagonal, horizontal and vertical directions.

The design tool takes the output of the synthesis tool and places and routes the design.

## Clock/Control Distribution Network

Each iCE40 device has eight global inputs, two pins on each side of the device. Note that not all GBINs are available in all packages.

These global inputs can be used as high fanout nets, clock, reset or enable signals. The dedicated global pins are identified as GBIN[7:0] and the global buffers are identified as-GBUF[7:0]. These eight inputs may be used as general purpose I/O if they are not used to drive the clock nets. Global buffer GBUF7 in I/O Bank 3 also provides an optional direct LVDS25 or subLVDS differential clock input.

Table 2-2 lists the connections between a specific global buffer and the inputs on a PLB. All global buffers optionally connect to the PLB CLK input. Any four of the eight global buffers can drive logic inputs to a PLB. Even-numbered global buffers optionally drive the Set/Reset input to a PLB. Similarly, odd-numbered buffers optionally drive the PLB clock-enable input.

***Table 2-2. Global Buffer (GBUF) Connections to Programmable Logic Blocks***

| Global Buffer | LUT Inputs | Clock | Clock Enable | Reset |
|---|---|---|---|---|
| GBUF0 | Yes, any 4 of 8 GBUF Inputs | ✓ | ✓ | |
| GBUF1 | | ✓ | | ✓ |
| GBUF2 | | ✓ | ✓ | |
| GBUF3 | | ✓ | | ✓ |
| GBUF4 | | ✓ | ✓ | |
| GBUF5 | | ✓ | | ✓ |
| GBUF6 | | ✓ | ✓ | |
| GBUF7 | | ✓ | | ✓ |

The maximum frequency for the global buffers are shown in the iCE40 External Switching Characteristics tables later in this document.

### Global Hi-Z Control

The global high-impedance control signal, GHIZ, connects to all I/O pins on the iCE40 device. This GHIZ signal is automatically asserted throughout the configuration process, forcing all user I/O pins into their high-impedance state.

### Global Reset Control

The global reset control signal connects to all PLB and PIO flip-flops on the iCE40 device. The global reset signal is automatically asserted throughout the configuration process, forcing all flip-flops to their defined wake-up state. For PLB flip-flops, the wake-up state is always reset, regardless of the PLB flip-flop primitive used in the application.

## sysCLOCK Phase Locked Loops (PLLs)

The sysCLOCK PLLs provide the ability to synthesize clock frequencies. The iCE40 devices have one or more sysCLOCK PLLs. REFERENCECLK is the reference frequency input to the PLL and its source can come from an external I/O pin or from internal routing. EXTFEEDBACK is the feedback signal to the PLL which can come from internal routing or an external I/O pin. The feedback divider is used to multiply the reference frequency and thus synthesize a higher frequency clock output.

The PLLOUT output has an output divider, thus allowing the PLL to generate different frequencies for each output. The output divider can have a value from 1 to 6. The PLLOUT outputs can all be used to drive the iCE40 global clock network directly or general purpose routing resources can be used.

The LOCK signal is asserted when the PLL determines it has achieved lock and de-asserted if a loss of lock is detected. A block diagram of the PLL is shown in Figure 2-3.

The timing of the device registers can be optimized by programming a phase shift into the PLLOUT output clock which will advance or delay the output clock with reference to the REFERENCECLK clock. This phase shift can be either programmed during configuration or can be adjusted dynamically. In dynamic mode, the PLL may lose lock after a phase adjustment on the output used as the feedback source and not relock until the $t_{LOCK}$ parameter has been satisfied.

For more details on the PLL, see TN1251, iCE40 sysCLOCK PLL Design and Usage Guide.

***Figure 2-3. PLL Diagram***

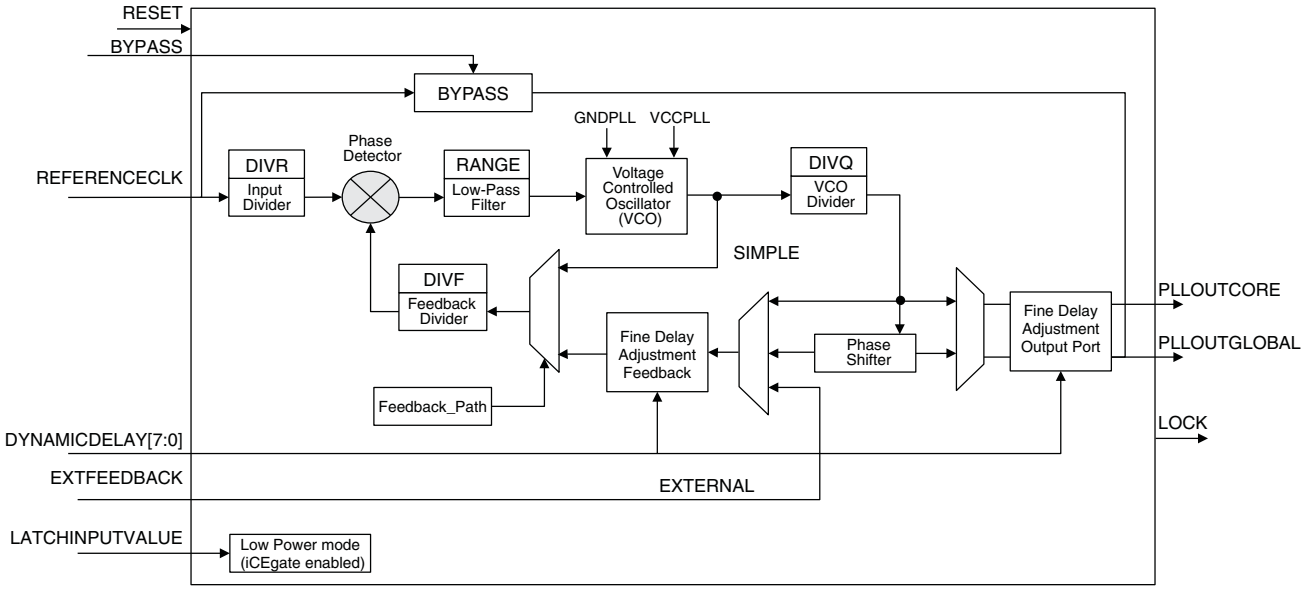



Table 2-3 provides signal descriptions of the PLL block.

*Table 2-3. PLL Signal Descriptions*

| Signal Name | Direction | Description |
|---|---|---|
| REFERENCECLK | Input | Input reference clock |
| BYPASS | Input | When FEEDBACK_PATH is set to SIMPLE, the BYPASS control selects which clock signal connects to the PLLOUT output.<br>0 = PLL generated signal<br>1 = REFERENCECLK |
| EXTFEEDBACK | Input | External feedback input to PLL. Enabled when the FEEDBACK_PATH attribute is set to EXTERNAL. |
| DYNAMICDELAY[3:0] | Input | Fine delay adjustment control inputs. Enabled when DELAY_ADJUSTMENT_MODE is set to DYNAMIC. |
| LATCHINPUTVALUE | Input | When enabled, forces the PLL into low-power mode; PLL output is held static at the last input clock value. Set ENABLE ICEGATE_PORTA and PORTB to '1' to enable. |
| PLLOUTGLOBAL | Output | Output from the Phase-Locked Loop (PLL). Drives a global clock network on the FPGA. The port has optimal connections to global clock buffers GBUF4 and GBUF5. |
| PLLOUTCORE | Output | Output clock generated by the PLL, drives regular FPGA routing. The frequency generated on this output is the same as the frequency of the clock signal generated on the PLLOUTLGOBAL port. |
| LOCK | Output | When High, indicates that the PLL output is phase aligned or locked to the input reference clock. |
| RESET | Input | Active low reset. |

## sysMEM Embedded Block RAM Memory

Larger iCE40 device includes multiple high-speed synchronous sysMEM Embedded Block RAMs (EBRs), each 4 Kbit in size. This memory can be used for a wide variety of purposes including data buffering, and FIFO.

### sysMEM Memory Block

The sysMEM block can implement single port, pseudo dual port, or FIFO memories with programmable logic resources. Each block can be used in a variety of depths and widths as shown in Table 2-4.

*Table 2-4. sysMEM Block Configurations[1]*

| Block RAM Configuration | Block RAM Configuration and Size | WADDR Port Size (Bits) | WDATA Port Size (Bits) | RADDR Port Size (Bits) | RDATA Port Size (Bits) | MASK Port Size (Bits) |
|---|---|---|---|---|---|---|
| SB_RAM256x16<br>SB_RAM256x16NR<br>SB_RAM256x16NW<br>SB_RAM256x16NRNW | 256x16 (4K) | 8 [7:0] | 16 [15:0] | 8 [7:0] | 16 [15:0] | 16 [15:0] |
| SB_RAM512x8<br>SB_RAM512x8NR<br>SB_RAM512x8NW<br>SB_RAM512x8NRNW | 512x8 (4K) | 9 [8:0] | 8 [7:0] | 9 [8:0] | 8 [7:0] | No Mask Port |
| SB_RAM1024x4<br>SB_RAM1024x4NR<br>SB_RAM1024x4NW<br>SB_RAM1024x4NRNW | 1024x4 (4K) | 10 [9:0] | 4 [3:0] | 10 [9:0] | 4 [3:0] | No Mask Port |
| SB_RAM2048x2<br>SB_RAM2048x2NR<br>SB_RAM2048x2NW<br>SB_RAM2048x2NRNW | 2048x2 (4K) | 11 [10:0] | 2 [1:0] | 11 [10:0] | 2 [1:0] | No Mask Port |

1. For iCE40 EBR primitives with a negative-edged Read or Write clock, the base primitive name is appended with a 'N' and a 'R' or 'W' depending on the clock that is affected.

### RAM Initialization and ROM Operation

If desired, the contents of the RAM can be pre-loaded during device configuration.

By preloading the RAM block during the chip configuration cycle and disabling the write controls, the sysMEM block can also be utilized as a ROM.

Note the sysMEM Embedded Block RAM Memory address 0 cannot be initialized.

### Memory Cascading

Larger and deeper blocks of RAM can be created using multiple EBR sysMEM Blocks.

### RAM4k Block

Figure 2-4 shows the 256x16 memory configurations and their input/output names. In all the sysMEM RAM modes, the input data and addresses for the ports are registered at the input of the memory array.

***Figure 2-4. sysMEM Memory Primitives***

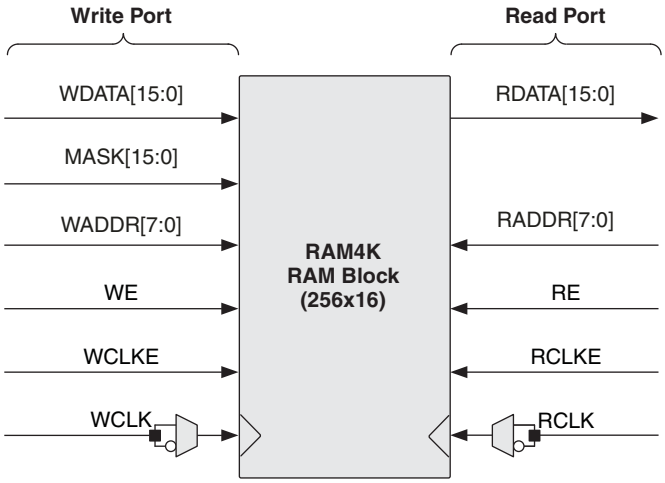



***Table 2-5. EBR Signal Descriptions***

| Signal Name | Direction | Description |
|---|---|---|
| WDATA[15:0] | Input | Write Data input. |
| MASK[15:0] | Input | Masks write operations for individual data bit-lines.<br>0 = write bit; 1 = don't write bit |
| WADDR[7:0] | Input | Write Address input. Selects one of 256 possible RAM locations. |
| WE | Input | Write Enable input. |
| WCLKE | Input | Write Clock Enable input. |
| WCLK | Input | Write Clock input. Default rising-edge, but with falling-edge option. |
| RDATA[15:0] | Output | Read Data output. |
| RADDR[7:0] | Input | Read Address input. Selects one of 256 possible RAM locations. |
| RE | Input | Read Enable input. |
| RCLKE | Input | Read Clock Enable input. |
| RCLK | Input | Read Clock input. Default rising-edge, but with falling-edge option. |

For further information on the sysMEM EBR block, please refer to TN1250, Memory Usage Guide for iCE40 Devices.

## sysIO

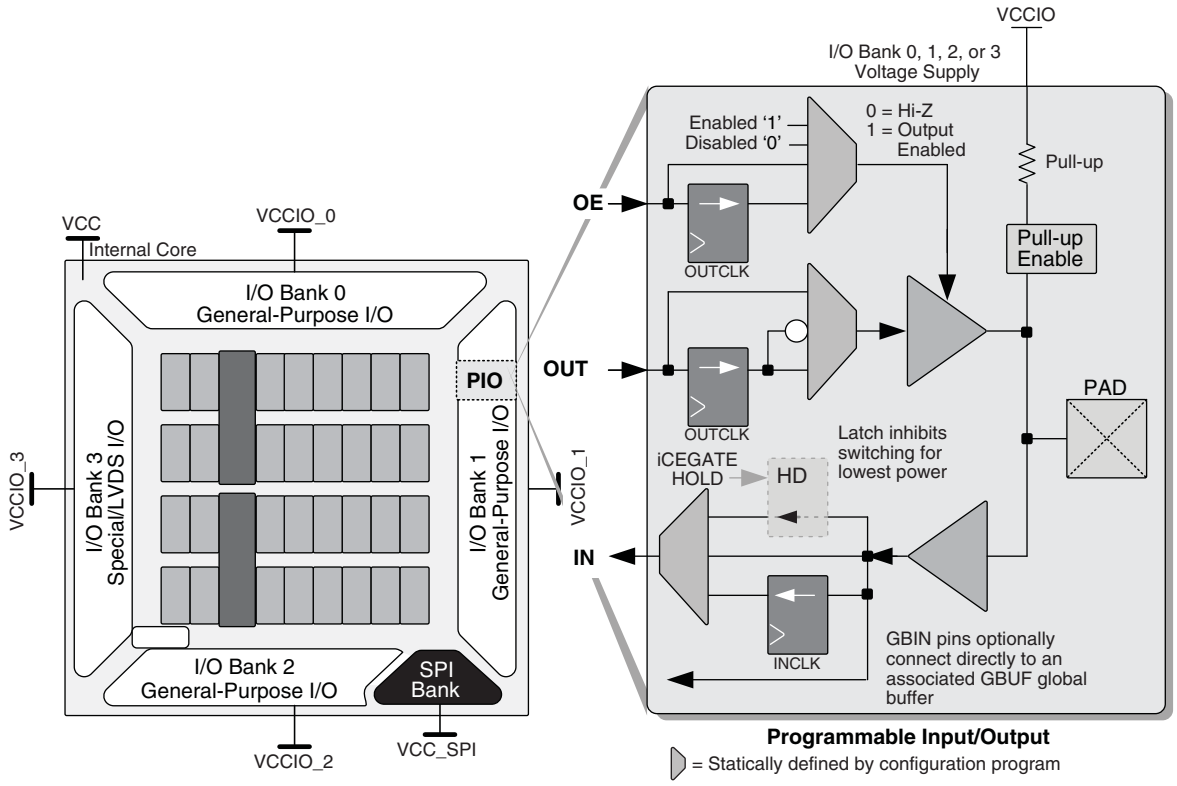
### Buffer Banks

iCE40 devices have up to four I/O banks with independent Vccio rails with an additional configuration bank $V_{CC\_SPI}$ for the SPI I/Os.

### Programmable I/O (PIO)

The programmable logic associated with an I/O is called a PIO. The individual PIO are connected to their respective sysIO buffers and pads. The PIOs are placed on all four sides of the device.

***Figure 2-5. I/O Bank and Programmable I/O Cell***




The PIO contains three blocks: an input register block, output register block iCEgate™ and tri-state register block. To save power, the optional iCEgate™ latch can selectively freeze the state of individual, non-registered inputs within an I/O bank. Note that the freeze signal is common to the bank. These blocks can operate in a variety of modes along with the necessary clock and selection logic.

### Input Register Block

The input register blocks for the PIOs on all edges contain registers that can be used to condition high-speed interface signals before they are passed to the device core. In Generic DDR mode, two registers are used to sample the data on the positive and negative edges of the system clock signal, creating two data streams.

### Output Register Block

The output register block can optionally register signals from the core of the device before they are passed to the sysIO buffers. In Generic DDR mode, two registers are used to capture the data on the positive and negative edge of the system clock and then muxed creating one data stream.

Figure 2-6 shows the input/output register block for the PIOs.

*Figure 2-6. iCE I/O Register Block Diagram*

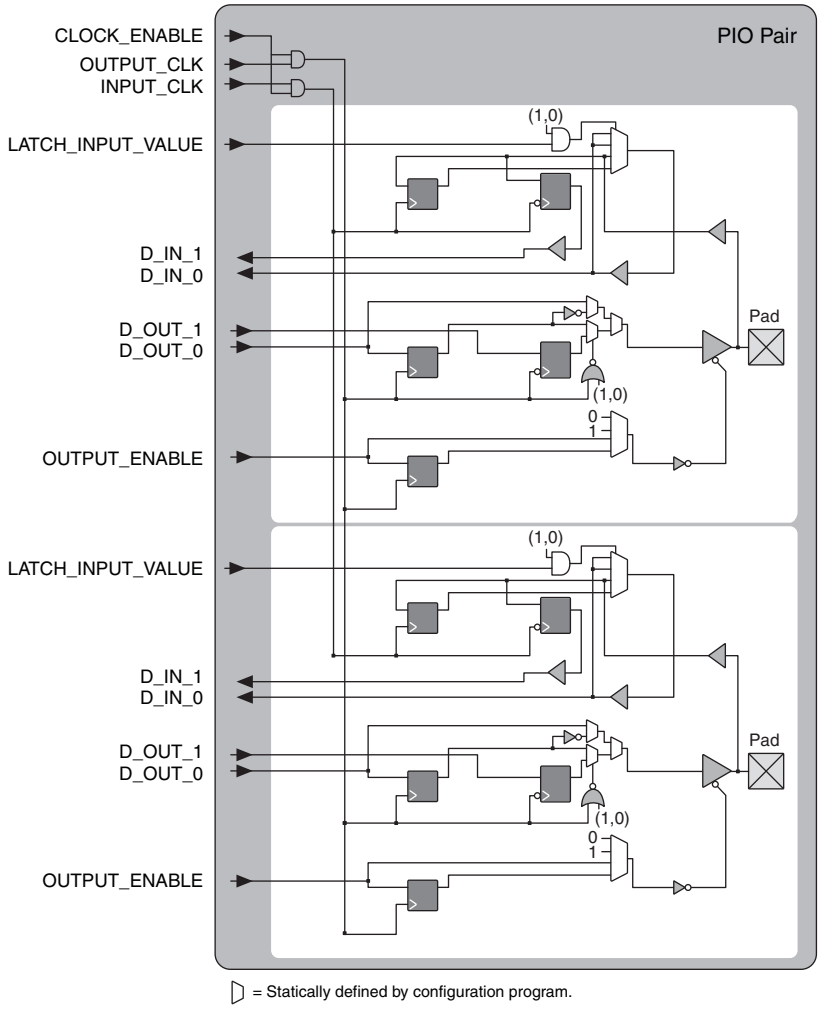



*Table 2-6. PIO Signal List*

| Pin Name | I/O Type | Description |
|---|---|---|
| OUTPUT_CLK | Input | Output register clock |
| CLOCK_ENABLE | Input | Clock enable |
| INPUT_CLK | Input | Input register clock |
| OUTPUT_ENABLE | Input | Output enable |
| D_OUT_0/1 | Input | Data from the core |
| D_IN_0/1 | Output | Data to the core |
| LATCH_INPUT_VALUE | Input | Latches/holds the Input Value |

## sysIO Buffer

Each I/O is associated with a flexible buffer referred to as a sysIO buffer. These buffers are arranged around the periphery of the device in groups referred to as banks. The sysIO buffers allow users to implement a wide variety of standards that are found in today's systems including LVCMOS and LVDS25.

High Current LED Drivers combine three sysIO buffers together. This allows for programmable drive strength. This also allows for high current drivers that are ideal to drive three white LEDs, or one RGB LED. Each bank is capable of supporting multiple I/O standards including single-ended LVCMOS buffers and differential LVDS25E output buf-

fers. Bank 3 additionally supports differential LVDS25 input buffers. Each sysIO bank has its own dedicated power supply.

**Typical I/O Behavior During Power-up**

The internal power-on-reset (POR) signal is deactivated when $V_{CC}$, $V_{CCIO\_2}$, $V_{PP\_2V5}$, and $V_{CC\_SPI}$ have reached the level defined in the Power-On-Reset Voltage table in the DC and Switching Characteristics section of this data sheet. After the POR signal is deactivated, the FPGA core logic becomes active. It is the user's responsibility to ensure that all $V_{CCIO}$ banks are active with valid input logic levels to properly control the output logic states of all the I/O banks that are critical to the application. The default configuration of the I/O pins in a device prior to configuration is tri-stated with a weak pull-up to $V_{CCIO}$. The I/O pins will maintain the pre-configuration state until $V_{CC}$ and $V_{CCIO}$ (for I/O banks containing configuration I/Os) have reached levels, at which time the I/Os will take on the software user-configured settings only after a proper download/configuration. Unused IOs are automatically blocked and the pullup termination is disabled.

**Supported Standards**

The iCE40 sysIO buffer supports both single-ended and differential input standards. The single-ended standard supported is LVCMOS. The buffer supports the LVCMOS 1.8, 2.5, and 3.3V standards. The buffer has individually configurable options for bus maintenance (weak pull-up or none).

Table 2-7 and Table 2-8 show the I/O standards (together with their supply and reference voltages) supported by the iCE40 devices.

***Table 2-7. Supported Input Standards***

| Input Standard | $V_{CCIO}$ (Typical) | | |
|---|---|---|---|
| | 3.3V | 2.5V | 1.8V |
| **Single-Ended Interfaces** | | | |
| LVCMOS33 | ✓ | | |
| LVCMOS25 | | ✓ | |
| LVCMOS18 | | | ✓ |
| **Differential Interfaces** | | | |
| LVDS25[1] | | ✓ | |
| subLVDS[1] | | | ✓ |

1. Bank 3 only.

***Table 2-8. Supported Output Standards***

| Output Standard | $V_{CCIO}$ (Typical) |
|---|---|
| **Single-Ended Interfaces** | |
| LVCMOS33 | 3.3 |
| LVCMOS25 | 2.5 |
| LVCMOS18 | 1.8 |
| **Differential Interfaces** | |
| LVDS25E[1] | 2.5 |
| subLVDSE[1] | 1.8 |

1. These interfaces can be emulated with external resistors in all devices.

## Non-Volatile Configuration Memory

All iCE40 devices provide a Non-Volatile Configuration Memory (NVCM) block which can be used to configure the device.

For more information on the NVCM, please refer to TN1248, iCE40 Programming and Configuration Usage Guide.

## Power On Reset

iCE40 devices have power-on reset circuitry to monitor $V_{CC}$, $V_{CCIO\_2}$, $V_{PP\_2V5}$, and $V_{CC\_SPI}$ voltage levels during power-up and operation. At power-up, the POR circuitry monitors $V_{CC}$, $V_{CCIO\_2}$, $V_{PP\_2V5}$, and $V_{CC\_SPI}$ (controls configuration) voltage levels. It then triggers download from the on-chip NVCM or external Flash memory after reaching the power-up levels specified in the Power-On-Reset Voltage table in the DC and Switching Characteristics section of this data sheet. Before and during configuration, the I/Os are held in tri-state. I/Os are released to user functionality once the device has finished configuration.

# Programming and Configuration

This section describes the programming and configuration of the iCE40 family.

### Device Programming

The NVCM memory can be programmed through the SPI port.

### Device Configuration

There are various ways to configure the Configuration RAM (CRAM) including:

1. Internal NVCM Download
2. From a SPI Flash (Master SPI mode)
3. System microprocessor to drive a Serial Slave SPI port (SSPI mode)

The image to configure the CRAM can be selected by the user on power up (Cold Boot) or once powered up (Warm Boot).

For more details on programming and configuration, see TN1248, iCE40 Programming and Configuration Usage Guide.

## Power Saving Options

iCE40 devices are available in two options for maximum flexibility: LP and HX devices. The LP devices have ultra low static and dynamic power consumption. HX devices are designed to provide high performance. Both the LP and the HX devices operate at 1.2V $V_{CC}$.

iCE40 devices feature iCEGate and PLL low power mode to allow users to meet the static and dynamic power requirements of their applications. While these features are available in both device types, these features are mainly intended for use with iCE40 LP devices to manage power consumption.

***Table 2-9. iCE40 Power Saving Features Description***

| Device Subsystem | Feature Description |
|---|---|
| PLL | When LATCHINPUTVALUE is enabled, forces the PLL into low-power mode; PLL output held static at last input clock value. |
| iCEGate | To save power, the optional iCEgate latch can selectively freeze the state of individual, non-registered inputs within an I/O bank. Registered inputs are effectively frozen by their associated clock or clock-enable control. |

# iCE40 LP/HX Family Data Sheet DC and Switching Characteristics

October 2013 Data Sheet DS1040

## Absolute Maximum Ratings[1, 2, 3, 4]

| | iCE40 LP/HX |
|---|---|
| Supply Voltage $V_{CC}$ | -0.5 to 1.42V |
| Output Supply Voltage $V_{CCIO}$, $V_{CC\_SPI}$ | -0.5 to 3.60V |
| NVCM Supply Voltage $V_{PP\_2V5}$ | -0.5 to 3.60V |
| PLL Supply Voltage $V_{CCPLL}$ | -0.5 to 1.30V |
| I/O Tri-state Voltage Applied | -0.5 to 3.60V |
| Dedicated Input Voltage Applied | -0.5 to 3.60V |
| Storage Temperature (Ambient) | -65°C to 150°C |
| Junction Temperature ($T_J$) | -55°C to 125°C |

1. Stress above those listed under the “Absolute Maximum Ratings” may cause permanent damage to the device. Functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied.
2. Compliance with the Lattice Thermal Management document is required.
3. All voltages referenced to GND.
4. IOs can support a 200mV Overshoot above the Recommend Operating Conditions $V_{CCIO}$ (Max) and -200mV Undershoot below $V_{IL}$ (Min). Overshoot and Undershoot is permitted for 25% duty cycle but must not exceed 1.6ns.

## Recommended Operating Conditions[1]

| Symbol | Parameter | | Min. | Max. | Units |
|---|---|---|---|---|---|
| $V_{CC}$[1] | Core Supply Voltage | | 1.14 | 1.26 | V |
| $V_{PP\_2V5}$ | $V_{PP\_2V5}$ NVCM Programming and Operating Supply Voltage | Slave SPI Configuration | 1.71 | 3.46 | V |
| | | Master SPI Configuration | 2.30 | 3.46 | V |
| | | Configure from NVCM | 2.30 | 3.46 | V |
| | | NVCM Programming | 2.30 | 3.00 | V |
| $V_{PP\_FAST}$[4] | Optional fast NVCM programming supply. Leave unconnected. | | N/A | N/A | V |
| $V_{CCPLL}$[5, 6] | PLL Supply Voltage | | 1.14 | 1.26 | V |
| $V_{CCIO}$[1, 2, 3] | I/O Driver Supply Voltage | $V_{CCIO0-3}$ | 1.71 | 3.46 | V |
| | | $V_{CC\_SPI}$ | 1.71 | 3.46 | V |
| $t_{JIND}$ | Junction Temperature Industrial Operation | | -40 | 100 | °C |
| $t_{PROG}$ | Junction Temperature NVCM Programming | | 10 | 30 | °C |

1. Like power supplies must be tied together. For example, if $V_{CCIO}$ and $V_{CC\_SPI}$ are both the same voltage, they must also be the same supply.
2. See recommended voltages by I/O standard in subsequent table.
3. $V_{CCIO}$ pins of unused I/O banks should be connected to the $V_{CC}$ power supply on boards.
4. $V_{PP\_FAST}$, used only for fast production programming, must be left floating or unconnected in applications, except CM36 and CM49 packages MUST have the $V_{PP\_FAST}$ ball connected to $V_{CCIO\_0}$ ball externally.
5. No PLL available on the iCE40LP384 and iCE40LP640 device.
6. $V_{CCPLL}$ is tied to $V_{CC}$ internally in packages without PLLs pins.

## Power Supply Ramp Rates[1, 2]

| Symbol | Parameter | | Min. | Max. | Units |
|---|---|---|---|---|---|
| $t_{RAMP}$ | Power supply ramp rates for all power supplies. | All configuration modes. No power supply sequencing. | 0.40 | 10 | V/ms |
| | | Configuring from Slave SPI. No power supply sequencing, | 0.01 | 10 | V/ms |
| | | Configuring from NVCM. $V_{CC}$ and $V_{PP\_2V5}$ to be powered 0.25ms before $V_{CC\_SPI}$. | 0.01 | 10 | V/ms |
| | | Configuring from MSPI. $V_{CC}$ and $V_{PP\_SPI}$ to be powered 0.25ms before $V_{PP\_2V5}$. | 0.01 | 10 | V/ms |

1. Assumes monotonic ramp rates.
2. iCE40LP384 requires $V_{CC}$ to be greater than 0.7V when $V_{CCIO}$ and $V_{CC\_SPI}$ are above GND.

## Power-On-Reset Voltage Levels[1]

| Symbol | Device | Parameter | | Min. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{PORUP}$ | iCE40LP384 | Power-On-Reset ramp-up trip point (band gap based circuit monitoring VCC, VCCIO_2, VCC_SPI and VPP_2V5) | VCC | 0.67 | 0.99 | V |
| | | | VCCIO_2 | 0.70 | 1.59 | V |
| | | | VCC_SPI | 0.70 | 1.59 | V |
| | | | VPP_2V5 | 0.70 | 1.59 | V |
| | iCE40LP640, iCE40LP/HX1K, iCE40LP/HX4K, iCE40LP/HX8K | Power-On-Reset ramp-up trip point (band gap based circuit monitoring VCC, VCCIO_2, VCC_SPI and VPP_2V5) | VCC | 0.55 | 0.75 | V |
| | | | VCCIO_2 | 0.86 | 1.29 | V |
| | | | VCC_SPI | 0.86 | 1.29 | V |
| | | | VPP_2V5 | 0.86 | 1.33 | V |
| $V_{PORDN}$ | iCE40LP384 | Power-On-Reset ramp-down trip point (band gap based circuit monitoring VCC, VCCIO_2, VCC_SPI and VPP_2V5) | VCC | - | 0.64 | V |
| | | | VCCIO_2 | - | 1.59 | V |
| | | | VCC_SPI | - | 1.59 | V |
| | | | VPP_2V5 | - | 1.59 | V |
| | iCE40LP640, iCE40LP/HX1K, iCE40LP/HX4K, iCE40LP/HX8K | Power-On-Reset ramp-down trip point (band gap based circuit monitoring VCC, VCCIO_2, VCC_SPI and VPP_2V5) | VCC | - | 0.75 | V |
| | | | VCCIO_2 | - | 1.29 | V |
| | | | VCC_SPI | - | 1.29 | V |
| | | | VPP_2V5 | - | 1.33 | V |

1. These POR trip points are only provided for guidance. Device operation is only characterized for power supply voltages specified under recommended operating conditions.

## ESD Performance

Please refer to the iCE40 Product Family Qualification Summary for complete qualification data, including ESD performance.

# DC Electrical Characteristics

**Over Recommended Operating Conditions**

| Symbol | Parameter | Condition | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $I_{IL}$, $I_{IH}$ [1, 3, 4, 5, 6, 7] | Input or I/O Leakage | $0V < V_{IN} < V_{CCIO} + 0.2V$ | — | — | +/-10 | µA |
| $C_1$ [6, 7] | I/O Capacitance[2] | $V_{CCIO}$ = 3.3V, 2.5V, 1.8V<br>$V_{CC}$ = Typ., $V_{IO}$ = 0 to $V_{CCIO}$ + 0.2V | — | 6 | — | pf |
| $C_2$ [6, 7] | Global Input Buffer Capacitance[2] | $V_{CCIO}$ = 3.3V, 2.5V, 1.8V<br>$V_{CC}$ = Typ., $V_{IO}$ = 0 to $V_{CCIO}$ + 0.2V | — | 6 | — | pf |
| $V_{HYST}$ | Input Hysteresis | $V_{CCIO}$ = 1.8V, 2.5V, 3.3V | — | 200 | — | mV |
| $I_{PU}$ [6, 7] | Internal PIO Pull-up Current | $V_{CCIO}$ = 1.8V, 0=<$V_{IN}$<=0.65 $V_{CCIO}$ | -3 | — | -31 | µA |
| | | $V_{CCIO}$ = 2.5V, 0=<$V_{IN}$<=0.65 $V_{CCIO}$ | -8 | — | -72 | µA |
| | | $V_{CCIO}$ = 3.3V, 0=<$V_{IN}$<=0.65 $V_{CCIO}$ | -11 | — | -128 | µA |

1. Input or I/O leakage current is measured with the pin configured as an input or as an I/O with the output driver tri-stated. It is not measured with the output driver active. Internal pull-up resistors are disabled.
2. $T_J$ 25°C, f = 1.0 MHz.
3. Please refer to $V_{IL}$ and $V_{IH}$ in the sysIO Single-Ended DC Electrical Characteristics table of this document.
4. Only applies to IOs in the SPI bank following configuration.
5. Some products are clamped to a diode when $V_{IN}$ is larger than $V_{CCIO}$.
6. High current IOs has three sysIO buffers connected together.
7. The iCE40640KLP and iCE401KLP SWG16 package has CDONE and a sysIO buffer are connected together.

# Static Supply Current – LP Devices[1, 2, 3, 4, 7]

| Symbol | Parameter | Device | Typ. $V_{CC}$[4] | Units |
|---|---|---|---|---|
| $I_{CC}$ | Core Power Supply | iCE40LP384 | 21 | µA |
| | | iCE40LP640 | 100 | µA |
| | | iCE40LP1K | 100 | µA |
| | | iCE40LP4K | 360 | µA |
| | | iCE40LP8K | 360 | µA |
| $I_{CCPLL}$ [5, 6] | PLL Power Supply | All devices | 10 | µA |
| $I_{PP\_2V5}$ | NVCM Power Supply | All devices | | µA |
| $I_{CCIO}$, $I_{CC\_SPI}$ | Bank Power Supply[4]<br>$V_{CCIO}$ = 2.5V | All devices | | µA |

1. Assumes blank pattern with the following characteristics: all outputs are tri-stated, all inputs are configured as LVCMOS and held at $V_{CCIO}$ or GND, on-chip PLL is off. For more detail with your specific design, use the Power Calculator tool. Power specified with master SPI configuration mode. Other modes may be up to 25% higher.
2. Frequency = 0 MHz.
3. $T_J$ = 25°C, power supplies at nominal voltage.
4. Does not include pull-up.
5. No PLL available on the iCE40LP384 and iCE40LP640 device.
6. $V_{CCPLL}$ is tied to $V_{CC}$ internally in packages without PLLs pins.
7. iCE40LP4K/iCE40LP8K status is Advanced.

## Static Supply Current – HX Devices[1, 2, 3, 4, 6]

| Symbol | Parameter | Device | Typ. $V_{CC}$[4] | Units |
|---|---|---|---|---|
| $I_{CC}$ | Core Power Supply | iCE40HX1K | 296 | µA |
| | | iCE40HX4K | 667 | µA |
| | | iCE40HX8K | 1100 | µA |
| $I_{CCPLL}$[5] | PLL Power Supply | All devices | 25 | µA |
| $I_{PP\_2V5}$ | NVCM Power Supply | All devices | | µA |
| $I_{CCIO}$, $I_{CC\_SPI}$ | Bank Power Supply[4] $V_{CCIO}$ = 2.5V | All devices | | µA |

1. Assumes blank pattern with the following characteristics: all outputs are tri-stated, all inputs are configured as LVCMOS and held at $V_{CCIO}$ or GND, on-chip PLL is off. For more detail with your specific design, use the Power Calculator tool. Power specified with master SPI configuration mode. Other modes may be up to 25% higher.
2. Frequency = 0 MHz.
3. $T_J$ = 25°C, power supplies at nominal voltage.
4. Does not include pull-up.
5. $V_{CCPLL}$ is tied to $V_{CC}$ internally in packages without PLLs pins.
6. iCE40HX4K/iCE40HX8K status is Advanced.

## Programming NVCM Supply Current – LP Devices[1, 2, 3, 4, 9]

| Symbol | Parameter | Device | Typ. $V_{CC}$[5] | Units |
|---|---|---|---|---|
| $I_{CC}$ | Core Power Supply | iCE40LP384 | 60 | µA |
| | | iCE40LP640 | 120 | µA |
| | | iCE40LP1K | 120 | µA |
| | | iCE40LP4K | | µA |
| | | iCE40LP8K | | µA |
| $I_{CCPLL}$[6, 7] | PLL Power Supply | All devices | | µA |
| $I_{PP\_2V5}$ | NVCM Power Supply | All devices | | µA |
| $I_{CCIO}$[8], $I_{CC\_SPI}$ | Bank Power Supply[5] | All devices | | µA |

1. Assumes all inputs are held at $V_{CCIO}$ or GND and all outputs are tri-stated.
2. Typical user pattern.
3. SPI programming is at 8 MHz.
4. $T_J$ = 25°C, power supplies at nominal voltage.
5. Per bank. $V_{CCIO}$ = 2.5V. Does not include pull-up.
6. No PLL available on the iCE40-LP384 and iCE40-LP640 device.
7. $V_{CCPLL}$ is tied to $V_{CC}$ internally in packages without PLLs pins.
8. $V_{PP\_FAST}$, used only for fast production programming, must be left floating or unconnected in applications, except CM36 and CM49 packages MUST have the $V_{PP\_FAST}$ ball connected to $V_{CCIO\_0}$ ball externally.
9. iCE40LP4K/iCE40LP8K status is Advanced.

## Programming NVCM Supply Current – HX Devices[1, 2, 3, 4, 8]

| Symbol | Parameter | Device | Typ. $V_{CC}$[5] | Units |
|---|---|---|---|---|
| $I_{CC}$ | Core Power Supply | iCE40HX1K | 238 | µA |
| | | iCE40HX4K | | µA |
| | | iCE40HX8K | | µA |
| $I_{CCPLL}$[6] | PLL Power Supply | All devices | | µA |
| $I_{PP\_2V5}$ | NVCM Power Supply | All devices | | mA |
| $I_{CCIO}$[7], $I_{CC\_SPI}$ | Bank Power Supply[5] | All devices | | mA |

1. Assumes all inputs are held at $V_{CCIO}$ or GND and all outputs are tri-stated.
2. Typical user pattern.
3. SPI programming is at 8 MHz.
4. $T_J$ = 25°C, power supplies at nominal voltage.
5. Per bank. $V_{CCIO}$ = 2.5V. Does not include pull-up.
6. $V_{CCPLL}$ is tied to $V_{CC}$ internally in packages without PLLs pins.
7. $V_{PP\_FAST}$, used only for fast production programming, must be left floating or unconnected in applications.
8. iCE40HX1K/iCE40HX4K/iCE40HX8K status is Advanced.

## Peak Startup Supply Current – LP Devices

| Symbol | Parameter | Device | Max | Units |
|---|---|---|---|---|
| $I_{CCPEAK}$ | Core Power Supply | iCE40LP384 | 7.7 | mA |
| | | iCELP640 | 6.4 | mA |
| | | iCE40LP1K | 6.4 | mA |
| | | iCE40LP4K | 15.7 | mA |
| | | iCE40LP8K | 15.7 | mA |
| $I_{CCPLLPEAK}$[1, 2] | PLL Power Supply | iCE40LP1K | 1.5 | mA |
| | | iCELP640 | 1.5 | mA |
| | | iCE40LP4K | 8.0 | mA |
| | | iCE40LP8K | 8.0 | mA |
| $I_{PP\_2V5PEAK}$ | NVCM Power Supply | iCE40LP384 | 3.0 | mA |
| | | iCELP640 | 7.7 | mA |
| | | iCE40LP1K | 7.7 | mA |
| | | iCE40LP4K | 4.2 | mA |
| | | iCE40LP8K | 4.2 | mA |
| $I_{PP\_FASTPEAK}$[3] | NVCM Programming Supply | iCE40LP384 | 5.7 | mA |
| | | iCELP640 | 8.1 | mA |
| | | iCE40LP1K | 8.1 | mA |
| $I_{CCIOPEAK}$[4], $I_{CC\_SPIPEAK}$ | Bank Power Supply | iCE40LP384 | 8.4 | mA |
| | | iCELP640 | 3.3 | mA |
| | | iCE40LP1K | 3.3 | mA |
| | | iCE40LP4K | 8.2 | mA |
| | | iCE40LP8K | 8.2 | mA |

1. No PLL available on the iCE40LP384 and iCE40LP640 device.
2. $V_{CCPLL}$ is tied to $V_{CC}$ internally in packages without PLLs pins.
3. $V_{PP\_FAST}$, used only for fast production programming, must be left floating or unconnected in applications, except CM36 and CM49 packages MUST have the $V_{PP\_FAST}$ ball connected to $V_{CCIO\_0}$ ball externally.
4. iCE40LP384 requires $V_{CC}$ to be greater than 0.7V when $V_{CCIO}$ and $V_{CC\_SPI}$ are above GND.

## Peak Startup Supply Current – HX Devices

| Symbol | Parameter | Device | Max | Units |
|---|---|---|---|---|
| $I_{CCPEAK}$ | Core Power Supply | iCE40HX1K | 6.9 | mA |
| | | iCE40HX4K | 22.3 | mA |
| | | iCE40HX8K | 22.3 | mA |
| $I_{CCPLLPEAK}$[1] | PLL Power Supply | iCE40HX1K | 1.8 | mA |
| | | iCE40HX4K | 6.4 | mA |
| | | iCE40HX8K | 6.4 | mA |
| $I_{PP\_2V5PEAK}$ | NVCM Power Supply | iCE40HX1K | 2.8 | mA |
| | | iCE40HX4K | 4.1 | mA |
| | | iCE40HX8K | 4.1 | mA |
| $I_{CCIOPEAK}$, $I_{CC\_SPIPEAK}$ | Bank Power Supply | iCE40HX1K | 6.8 | mA |
| | | iCE40HX4K | 6.8 | mA |
| | | iCE40HX8K | 6.8 | mA |

1. $V_{CCPLL}$ is tied to $V_{CC}$ internally in packages without PLLs pins.

## sysIO Recommended Operating Conditions

| Standard | $V_{CCIO}$ (V) | | |
|---|---|---|---|
| | Min. | Typ. | Max. |
| LVCMOS 3.3 | 3.14 | 3.3 | 3.46 |
| LVCMOS 2.5 | 2.37 | 2.5 | 2.62 |
| LVCMOS 1.8 | 1.71 | 1.8 | 1.89 |
| LVDS25E[1, 2] | 2.37 | 2.5 | 2.62 |
| subLVDSE[1, 2] | 1.71 | 1.8 | 1.89 |

1. Inputs on-chip. Outputs are implemented with the addition of external resistors.
2. Does not apply to Configuration Bank $V_{CC\_SPI}$.

## sysIO Single-Ended DC Electrical Characteristics

| Input/ Output Standard | $V_{IL}$ | | $V_{IH}$[1] | | $V_{OL}$ Max. (V) | $V_{OH}$ Min. (V) | $I_{OL}$ Max. (mA) | $I_{OH}$ Max. (mA) |
|---|---|---|---|---|---|---|---|---|
| | Min. (V) | Max. (V) | Min. (V) | Max. (V) | | | | |
| LVCMOS 3.3 | -0.3 | 0.8 | 2.0 | $V_{CCIO}$ + 0.2V | 0.4 | $V_{CCIO}$ - 0.5 | 8, 16[2], 24[2] | -8, -16[2], -24[2] |
| | | | | | 0.2 | $V_{CCIO}$ - 0.2 | 0.1 | -0.1 |
| LVCMOS 2.5 | -0.3 | 0.7 | 1.7 | $V_{CCIO}$ + 0.2V | 0.4 | $V_{CCIO}$ - 0.5 | 6, 12[2], 18[2] | -6, -12[2], -18[2] |
| | | | | | 0.2 | $V_{CCIO}$ - 0.2 | 0.1 | -0.1 |
| LVCMOS 1.8 | -0.3 | $0.35V_{CCIO}$ | $0.65V_{CCIO}$ | $V_{CCIO}$ + 0.2V | 0.4 | $V_{CCIO}$ - 0.4 | 4, 8[2], 12[2] | -4, -8[2], -12[2] |
| | | | | | 0.2 | $V_{CCIO}$ - 0.2 | 0.1 | -0.1 |

1. Some products are clamped to a diode when $V_{IN}$ is larger than $V_{CCIO}$.
2. Only for High Drive LED outputs.

## sysIO Differential Electrical Characteristics

The LVDS25E/subLVDSE differential output buffers are available on all banks but the LVDS/subLVDS input buffers are only available on Bank 3 of iCE40 devices.

### LVDS25

**Over Recommended Operating Conditions**

| Parameter Symbol | Parameter Description | Test Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{INP}$, $V_{INM}$ | Input Voltage | $V_{CCIO}$[1] = 2.5 | 0 | — | 2.5 | V |
| $V_{THD}$ | Differential Input Threshold | | 250 | 350 | 450 | mV |
| $V_{CM}$ | Input Common Mode Voltage | $V_{CCIO}$[1] = 2.5 | ($V_{CCIO}$/2) - 0.3 | $V_{CCIO}$/2 | ($V_{CCIO}$/2) + 0.3 | V |
| $I_{IN}$ | Input Current | Power on | — | — | ±10 | µA |

1. Typical.

### subLVDS

**Over Recommended Operating Conditions**

| Parameter Symbol | Parameter Description | Test Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{INP}$, $V_{INM}$ | Input Voltage | $V_{CCIO}$[1] = 1.8 | 0 | — | 1.8 | V |
| $V_{THD}$ | Differential Input Threshold | | 100 | 150 | 200 | mV |
| $V_{CM}$ | Input Common Mode Voltage | $V_{CCIO}$[1] = 1.8 | ($V_{CCIO}$/2) - 0.25 | $V_{CCIO}$/2 | ($V_{CCIO}$/2) + 0.25 | V |
| $I_{IN}$ | Input Current | Power on | — | — | ±10 | µA |

1. Typical.

## LVDS25E Emulation

iCE40 devices can support LVDSE outputs via emulation on all banks. The output is emulated using complementary LVCMOS outputs in conjunction with resistors across the driver outputs on all devices. The scheme shown in Figure 3-1 is one possible solution for LVDS25E standard implementation. Resistor values in Figure 3-1 are industry standard values for 1% resistors.

***Figure 3-1. LVDS25E Using External Resistors***

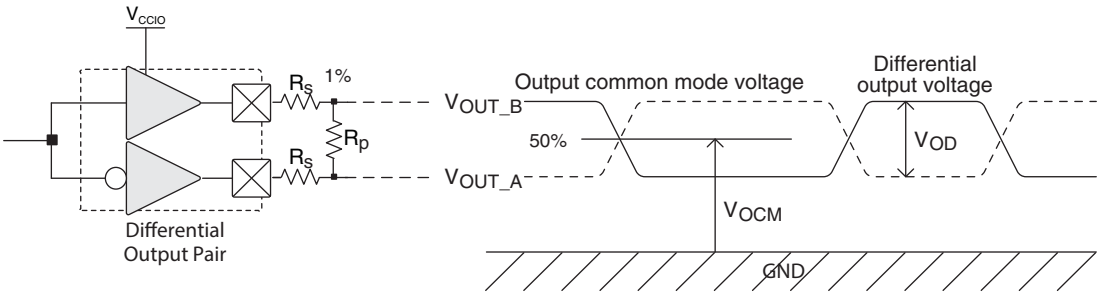



***Table 3-1. LVDS25E DC Conditions***

**Over Recommended Operating Conditions**

| Parameter | Description | Typ. | Units |
|---|---|---|---|
| $Z_{OUT}$ | Output impedance | 20 | Ohms |
| $R_S$ | Driver series resistor | 150 | Ohms |
| $R_P$ | Driver parallel resistor | 140 | Ohms |
| $R_T$ | Receiver termination | 100 | Ohms |
| $V_{OH}$ | Output high voltage | 1.43 | V |
| $V_{OL}$ | Output low voltage | 1.07 | V |
| $V_{OD}$ | Output differential voltage | 0.30 | V |
| $V_{CM}$ | Output common mode voltage | 1.25 | V |
| $Z_{BACK}$ | Back impedance | 100.5 | Ohms |
| $I_{DC}$ | DC output current | 6.03 | mA |

## SubLVDS Emulation

The iCE40 family supports the differential subLVDS standard. The output standard is emulated using complementary LVCMOS outputs in conjunction with resistors across the driver outputs on all banks of the devices. The subLVDS input standard is supported by the LVDS25 differential input buffer. The scheme shown in Figure 3-2 is one possible solution for subLVDSE output standard implementation. Use LVDS25E mode with suggested resistors for subLVDSE operation. Resistor values in Figure 3-2 are industry standard values for 1% resistors.

***Figure 3-2. subLVDSE***

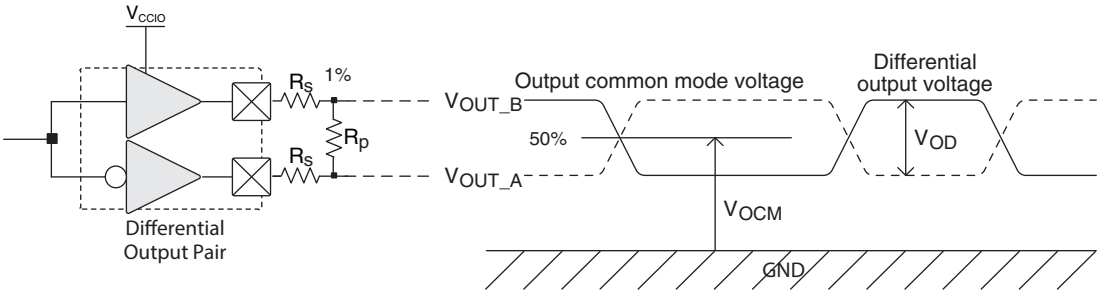



***Table 3-2. subLVDSE DC Conditions***

**Over Recommended Operating Conditions**

| Parameter | Description | Typ. | Units |
|---|---|---|---|
| $Z_{OUT}$ | Output impedance | 20 | Ohms |
| $R_S$ | Driver series resistor | 270 | Ohms |
| $R_P$ | Driver parallel resistor | 120 | Ohms |
| $R_T$ | Receiver termination | 100 | Ohms |
| $V_{OH}$ | Output high voltage | 1.43 | V |
| $V_{OL}$ | Output low voltage | 1.07 | V |
| $V_{OD}$ | Output differential voltage | 0.35 | V |
| $V_{CM}$ | Output common mode voltage | 0.9 | V |
| $Z_{BACK}$ | Back impedance | 100.5 | Ohms |
| $I_{DC}$ | DC output current | 2.8 | mA |

# Typical Building Block Function Performance – LP Devices[1, 2]

## Pin-to-Pin Performance (LVCMOS25)

| Function | Timing | Units |
|---|---|---|
| **Basic Functions** | | |
| 16-bit decoder | 11.0 | ns |
| 4:1 MUX | 12.0 | ns |
| 16:1 MUX | 13.0 | ns |

## Register-to-Register Performance

| Function | Timing | Units |
|---|---|---|
| **Basic Functions** | | |
| 16:1 MUX | 190 | MHz |
| 16-bit adder | 160 | MHz |
| 16-bit counter | 175 | MHz |
| 64-bit counter | 65 | MHz |
| **Embedded Memory Functions** | | |
| 256x16 Pseudo-Dual Port RAM | 240 | MHz |

1. The above timing numbers are generated using the iCECube2 design tool. Exact performance may vary with device and tool version. The tool uses internal parameters that have been characterized but are not tested on every device.
2. Using a $V_{CC}$ of 1.14V at Junction Temp 85C.

# Typical Building Block Function Performance – HX Devices[1, 2]

## Pin-to-Pin Performance (LVCMOS25)

| Function | Timing | Units |
|---|---|---|
| **Basic Functions** | | |
| 16-bit decoder | 10.0 | ns |
| 4:1 MUX | 9.0 | ns |
| 16:1 MUX | 9.5 | ns |

## Register-to-Register Performance

| Function | Timing | Units |
|---|---|---|
| **Basic Functions** | | |
| 16:1 MUX | 305 | MHz |
| 16-bit adder | 220 | MHz |
| 16-bit counter | 255 | MHz |
| 64-bit counter | 105 | MHz |
| **Embedded Memory Functions** | | |
| 256x16 Pseudo-Dual Port RAM | 403 | MHz |

1. The above timing numbers are generated using the iCECube2 design tool. Exact performance may vary with device and tool version. The tool uses internal parameters that have been characterized but are not tested on every device.
2. Using a $V_{CC}$ of 1.14V at Junction Temp 85C.

## Derating Logic Timing

Logic timing provided in the following sections of the data sheet and the Lattice design tools are worst case numbers in the operating range. Actual delays may be much faster. Lattice design tools can provide logic timing numbers at a particular temperature and voltage.

## Maximum sysIO Buffer Performance[2]

| I/O Standard | Max. Speed | Units |
|---|---|---|
| **Inputs** | | |
| LVDS25[1] | 400 | MHz |
| subLVDS18[1] | 400 | MHz |
| LVCMOS33 | 250 | MHz |
| LVCMOS25 | 250 | MHz |
| LVCMOS18 | 250 | MHz |
| **Outputs** | | |
| LVDS25E | 250 | MHz |
| subLVDS18E | 155 | MHz |
| LVCMOS33 | 250 | MHz |
| LVCMOS25 | 250 | MHz |
| LVCMOS18 | 155 | MHz |

1. Supported in Bank 3 only.
2. Measured with a toggling pattern

## iCE40 Family Timing Adders

**Over Recommended Commercial Operating Conditions - LP Devices[1, 2, 3, 4, 5]**

| Buffer Type | Description | Timing | Units |
|---|---|---|---|
| **Input Adjusters** | | | |
| LVDS25 | LVDS, $V_{CCIO}$ = 2.5V | -0.18 | ns |
| subLVDS | subLVDS, $V_{CCIO}$ = 1.8V | 0.82 | ns |
| LVCMOS33 | LVCMOS, $V_{CCIO}$ = 3.3V | 0.18 | ns |
| LVCMOS25 | LVCMOS, $V_{CCIO}$ = 2.5V | 0.00 | ns |
| LVCMOS18 | LVCMOS, $V_{CCIO}$ = 1.8V | 0.19 | ns |
| **Output Adjusters** | | | |
| LVDS25E | LVDS, Emulated, $V_{CCIO}$ = 2.5V | 0.00 | ns |
| subLVDSE | subLVDS, Emulated, $V_{CCIO}$ = 1.8V | 1.32 | ns |
| LVCMOS33 | LVCMOS, $V_{CCIO}$ = 3.3V | -0.12 | ns |
| LVCMOS25 | LVCMOS, $V_{CCIO}$ = 2.5V | 0.00 | ns |
| LVCMOS18 | LVCMOS, $V_{CCIO}$ = 1.8V | 1.32 | ns |

1. Timing adders are relative to LVCMOS25 and characterized but not tested on every device.
2. LVCMOS timing measured with the load specified in Switching Test Condition table.
3. All other standards tested according to the appropriate specifications.
4. Commercial timing numbers are shown.
5. Not all I/O standards are supported for all banks. See the Architecture section of this data sheet for details.

**Over Recommended Commercial Operating Conditions - HX Devices[1, 2, 3, 4, 5]**

| Buffer Type | Description | Timing | Units |
|---|---|---|---|
| **Input Adjusters** | | | |
| LVDS25 | LVDS, $V_{CCIO}$ = 2.5V | 0.13 | ns |
| subLVDS | subLVDS, $V_{CCIO}$ = 1.8V | 1.03 | ns |
| LVCMOS33 | LVCMOS, $V_{CCIO}$ = 3.3V | 0.16 | ns |
| LVCMOS25 | LVCMOS, $V_{CCIO}$ = 2.5V | 0.00 | ns |
| LVCMOS18 | LVCMOS, $V_{CCIO}$ = 1.8V | 0.23 | ns |
| **Output Adjusters** | | | |
| LVDS25E | LVDS, Emulated, $V_{CCIO}$ = 2.5V | 0.00 | ns |
| subLVDSE | subLVDS, Emulated, $V_{CCIO}$ = 1.8V | 1.76 | ns |
| LVCMOS33 | LVCMOS, $V_{CCIO}$ = 3.3V | 0.17 | ns |
| LVCMOS25 | LVCMOS, $V_{CCIO}$ = 2.5V | 0.00 | ns |
| LVCMOS18 | LVCMOS, $V_{CCIO}$ = 1.8V | 1.76 | ns |

1. Timing adders are relative to LVCMOS25 and characterized but not tested on every device.
2. LVCMOS timing measured with the load specified in Switching Test Condition table.
3. All other standards tested according to the appropriate specifications.
4. Commercial timing numbers are shown.
5. Not all I/O standards are supported for all banks. See the Architecture section of this data sheet for details.

# iCE40 External Switching Characteristics – LP Devices [1, 2]

**Over Recommended Operating Conditions**

| Parameter | Description | Device | Min. | Max. | Units |
|---|---|---|---|---|---|
| **Clocks** | | | | | |
| **Global Clocks** | | | | | |
| $f_{MAX\_GBUF}$ | Frequency for Global Buffer Clock network | All iCE40LP devices | — | 275 | MHz |
| $t_{W\_GBUF}$ | Clock Pulse Width for Global Buffer | All iCE40LP devices | 0.92 | — | ns |
| $t_{SKEW\_GBUF}$ | Global Buffer Clock Skew Within a Device | iCE40LP384 | — | 370 | ps |
| | | iCE40LP640 | — | 230 | ps |
| | | iCE40LP1K | — | 230 | ps |
| | | iCE40LP4K | — | 340 | ps |
| | | iCE40LP8K | — | 340 | ps |
| **Pin-LUT-Pin Propagation Delay** | | | | | |
| $t_{PD}$ | Best case propagation delay through one LUT-4 | All iCE40LP devices | — | 9.36 | ns |
| **General I/O Pin Parameters (Using Global Buffer Clock without PLL)[3]** | | | | | |
| $t_{SKEW\_IO}$ | Data bus skew across a bank of IOs | iCE40LP384 | — | 300 | ps |
| | | iCE40LP640 | — | 200 | ps |
| | | iCE40LP1K | — | 200 | ps |
| | | iCE40LP4K | — | 280 | ps |
| | | iCE40LP8K | — | 280 | ps |
| $t_{CO}$ | Clock to Output - PIO Output Register | iCE40LP384 | — | 6.33 | ns |
| | | iCE40LP640 | — | 5.91 | ns |
| | | iCE40LP1K | — | 5.91 | ns |
| | | iCE40LP4K | — | 6.58 | ns |
| | | iCE40LP8K | — | 6.58 | ns |
| $t_{SU}$ | Clock to Data Setup - PIO Input Register | iCE40LP384 | -0.08 | — | ns |
| | | iCE40LP640 | -0.33 | — | ns |
| | | iCE40LP1K | -0.33 | — | ns |
| | | iCE40LP4K | -0.63 | — | ns |
| | | iCE40LP8K | -0.63 | — | ns |
| $t_{H}$ | Clock to Data Hold - PIO Input Register | iCE40LP384 | 1.99 | — | ns |
| | | iCE40LP640 | 2.81 | — | ns |
| | | iCE40LP1K | 2.81 | — | ns |
| | | iCE40LP4K | 3.48 | — | ns |
| | | iCE40LP8K | 3.48 | — | ns |
| **General I/O Pin Parameters (Using Global Buffer Clock with PLL)[3]** | | | | | |
| $t_{COPLL}$ | Clock to Output - PIO Output Register | iCE40LP1K | — | 2.20 | ns |
| | | iCE40LP4K | — | 2.30 | ns |
| | | iCE40LP8K | — | 2.30 | ns |
| $t_{SUPLL}$ | Clock to Data Setup - PIO Input Register | iCE40LP1K | 5.23 | — | ns |
| | | iCE40LP4K | 6.13 | — | ns |
| | | iCE40LP8K | 6.13 | — | ns |

## iCE40 External Switching Characteristics – LP Devices (Continued)[1, 2]

**Over Recommended Operating Conditions**

| Parameter | Description | Device | Min. | Max. | Units |
|---|---|---|---|---|---|
| $t_{HPLL}$ | Clock to Data Hold - PIO Input Register | iCE40LP1K | -0.90 | — | ns |
| | | iCE40LP4K | -0.80 | — | ns |
| | | iCE40LP8K | -0.80 | — | ns |

1. Exact performance may vary with device and design implementation. Commercial timing numbers are shown at 85°C and 1.14V. Other operating conditions can be extracted from the iCECube2 software.
2. General I/O timing numbers based on LVCMOS 2.5, 0pf load.
3. Supported on devices with a PLL.

# iCE40 External Switching Characteristics – HX Devices [1, 2]

**Over Recommended Operating Conditions**

| Parameter | Description | Device | Min. | Max. | Units |
|---|---|---|---|---|---|
| **Clocks** | | | | | |
| **Primary Clocks** | | | | | |
| $f_{MAX\_GBUF}$ | Frequency for Global Buffer Clock network | All iCE40HX devices | — | 275 | MHz |
| $t_{W\_GBUF}$ | Clock Pulse Width for Global Buffer | All iCE40HX devices | 0.88 | — | ns |
| $t_{SKEW\_GBUF}$ | Global Buffer Clock Skew Within a Device | iCE40HX1K | — | 727 | ps |
| | | iCE40HX4K | — | 300 | ps |
| | | iCE40HX8K | — | 300 | ps |
| **Pin-LUT-Pin Propagation Delay** | | | | | |
| $t_{PD}$ | Best case propagation delay through one LUT-4 | All iCE40HX devices | — | 7.30 | ns |
| **General I/O Pin Parameters (Using Global Buffer Clock without PLL)** | | | | | |
| $t_{SKEW\_IO}$ | Data bus skew across a bank of IOs | iCE40HX1K | — | 696 | ps |
| | | iCE40HX4K | — | 290 | ps |
| | | iCE40HX8K | — | 290 | ps |
| $t_{CO}$ | Clock to Output - PIO Output Register | iCE40HX1K | — | 5.00 | ns |
| | | iCE40HX4K | — | 5.41 | ns |
| | | iCE40HX8K | — | 5.41 | ns |
| $t_{SU}$ | Clock to Data Setup - PIO Input Register | iCE40HX1K | -0.23 | — | ns |
| | | iCE40HX4K | -0.43 | — | ns |
| | | iCE40HX8K | -0.43 | — | ns |
| $t_{H}$ | Clock to Data Hold - PIO Input Register | iCE40HX1K | 1.92 | — | ns |
| | | iCE40HX4K | 2.38 | — | ns |
| | | iCE40HX8K | 2.38 | — | ns |
| **General I/O Pin Parameters (Using Global Buffer Clock with PLL)[3]** | | | | | |
| $t_{COPLL}$ | Clock to Output - PIO Output Register | iCE40HX1K | — | 2.96 | ns |
| | | iCE40HX4K | — | 2.51 | ns |
| | | iCE40HX8K | — | 2.51 | ns |
| $t_{SUPLL}$ | Clock to Data Setup - PIO Input Register | iCE40HX1K | 3.10 | — | ns |
| | | iCE40HX4K | 4.16 | — | ns |
| | | iCE40HX8K | 4.16 | — | ns |
| $t_{HPLL}$ | Clock to Data Hold - PIO Input Register | iCE40HX1K | -0.60 | — | ns |
| | | iCE40HX4K | -0.53 | — | ns |
| | | iCE40HX8K | -0.53 | — | ns |

1. Exact performance may vary with device and design implementation. Commercial timing numbers are shown at 85°C and 1.14V. Other operating conditions, including industrial, can be extracted from the iCECube2 software.
2. General I/O timing numbers based on LVCMOS 2.5, 0pf load.
3. Supported on devices with a PLL.

# sysCLOCK PLL Timing

**Over Recommended Operating Conditions**

| Parameter | Descriptions | Conditions | Min. | Max. | Units |
|---|---|---|---|---|---|
| $f_{IN}$ | Input Clock Frequency (REFERENCECLK, EXTFEEDBACK) | | 10 | 133 | MHz |
| $f_{OUT}$ | Output Clock Frequency (PLLOUT) | | 16 | 275 | MHz |
| $f_{VCO}$ | PLL VCO Frequency | | 533 | 1066 | MHz |
| $f_{PFD}$ | Phase Detector Input Frequency | | 10 | 133 | MHz |
| **AC Characteristics** | | | | | |
| $t_{DT}$ | Output Clock Duty Cycle | $f_{OUT}$ < 175MHz. Without duty trim selected | 40 | 50 | % |
| | | 175MHz < $f_{OUT}$ < 275 MHz. Without duty trim selected | 35 | 65 | "% |
| $t_{PH}$ | Output Phase Accuracy | | — | +/-12 | deg |
| $t_{OPJIT}$[1, 5] | Output Clock Period Jitter | $f_{OUT}$ <= 100 MHz | — | 450 | ps p-p |
| | | $f_{OUT}$ > 100 MHz | — | 0.05 | UIPP |
| | Output Clock Cycle-to-cycle Jitter | $f_{OUT}$ <= 100 MHz | — | 750 | ps p-p |
| | | $f_{OUT}$ > 100 MHz | — | 0.10 | UIPP |
| | Output Clock Phase Jitter | $f_{PFD}$ <= 25 MHz | — | 275 | ps p-p |
| | | $f_{PFD}$ > 25 MHz | — | 0.05 | UIPP |
| $t_W$ | Output Clock Pulse Width | At 90% or 10% | 1.3 | — | ns |
| $t_{LOCK}$[2, 3] | PLL Lock-in Time | | — | 50 | us |
| $t_{UNLOCK}$ | PLL Unlock Time | | — | 50 | ns |
| $t_{IPJIT}$[4] | Input Clock Period Jitter | $f_{PFD} \geq$ 20 MHz | — | 1000 | ps p-p |
| | | $f_{PFD}$ < 20 MHz | — | 0.02 | UIPP |
| $t_{FDTAP}$ | Fine Delay adjustment, per Tap | | 147 | 195 | ps |
| $t_{STABLE}$[3] | LATCHINPUTVALUE LOW to PLL Stable | | — | 500 | ns |
| $t_{STABLE\_PW}$[3] | LATCHINPUTVALUE Pulse Width | | — | 100 | ns |
| $t_{RST}$ | RESET Pulse Width | | 10 | — | ns |
| $t_{RSTREC}$ | RESET Recovery Time | | 10 | — | us |
| $t_{DYNAMIC\_WD}$ | DYNAMICDELAY Pulse Width | | 100 | — | VCO Cycles |
| $t_{PDBYPASS}$ | Propagation delay with the PLL in bypass mode | iCE40LP | 1.18 | 4.68 | ns |
| | | iCE40HX | 1.73 | 4.07 | ns |

1. Period jitter sample is taken over 10,000 samples of the primary PLL output with a clean reference clock. Cycle-to-cycle jitter is taken over 1000 cycles. Phase jitter is taken over 2000 cycles. All values per JESD65B.
2. Output clock is valid after $t_{LOCK}$ for PLL reset and dynamic delay adjustment.
3. At minimum $f_{PFD}$. As the $f_{PFD}$ increases the time will decrease to approximately 60% the value listed.
4. Maximum limit to prevent PLL unlock from occurring. Does not imply the PLL will operate within the output specifications listed in this table.
5. The jitter values will increase with loading of the PLD fabric and in the presence of SSO noise.

## SPI Master or NVCM Configuration Time[1, 2]

| Symbol | Parameter | Conditions | Typ. | Units |
|---|---|---|---|---|
| $t_{CONFIG}$ | POR/CRESET_B to Device I/O Active | iCE40LP384 - Low Frequency (Default) | 25 | ms |
| | | iCE40LP384 - Medium Frequency | 15 | ms |
| | | iCE40LP384 - High Frequency | 11 | ms |
| | | iCE40LP640 - Low Frequency (Default) | 53 | ms |
| | | iCE40LP640 - Medium Frequency | 25 | ms |
| | | iCE40LP640 - High Frequency | 13 | ms |
| | | iCE40LP/HX1K - Low Frequency (Default) | 53 | ms |
| | | iCE40LP/HX1K - Medium Frequency | 25 | ms |
| | | iCE40LP/HX1K - High Frequency | 13 | ms |
| | | iCE40LP/HX4K - Low Frequency (Default) | 230 | ms |
| | | iCE40LP/HX4K - Medium Frequency | 110 | ms |
| | | iCE40LP/HX4K - High Frequency | 70 | ms |
| | | iCE40LP/HX8K - Low Frequency (Default) | 230 | ms |
| | | iCE40LP/HX8K - Medium Frequency | 110 | ms |
| | | iCE40LP/HX8K - High Frequency | 70 | ms |

1. Assumes sysMEM Block is initialized to an all zero pattern if they are used.
2. The NVCM download time is measured with a fast ramp rate starting from the maximum voltage of POR trip point.

## sysCONFIG Port Timing Specifications[1]

| Symbol | Parameter | | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| **All Configuration Modes** | | | | | | |
| $t_{CRESET\_B}$ | Minimum CRESET_B Low pulse width required to restart configuration, from falling edge to rising edge | | 200 | — | — | ns |
| $t_{DONE\_IO}$ | Number of configuration clock cycles after CDONE goes High before the PIO pins are activated | | 49 | — | — | Clock Cycles |
| **Slave SPI** | | | | | | |
| $t_{CR\_SCK}$ | Minimum time from a rising edge on CRESET_B until the first SPI write operation, first SPI_SCK. During this time, the iCE40 device is clearing its internal configuration memory | iCE40LP384 | 600 | - | — | us |
| | | iCE40LP640, iCE40LP/HX1K | 800 | - | — | us |
| | | iCE40LP/HX4K | 1200 | - | — | us |
| | | iCE40LP/HX8K | 1200 | - | — | us |
| $f_{MAX}$[1] | CCLK clock frequency | Write | 1 | - | 25 | MHz |
| | | Read iCE40LP384[2] | - | 15 | - | MHz |
| | | Read iCE40LP640, iCE40LP/HX1K[2] | - | 15 | - | MHz |
| | | Read iCE40LP/HX4K[2] | - | 15 | - | MHz |
| | | Read iCE40LP/HX8K[2] | - | 15 | - | MHz |
| $t_{CCLKH}$ | CCLK clock pulse width high | | 20 | — | — | ns |
| $t_{CCLKL}$ | CCLK clock pulse width low | | 20 | — | — | ns |
| $t_{STSU}$ | CCLK setup time | | 12 | | — | ns |
| $t_{STH}$ | CCLK hold time | | 12 | — | — | ns |
| $t_{STCO}$ | CCLK falling edge to valid output | | 13 | — | — | ns |
| **Master SPI** | | | | | | |
| $f_{MCLK}$ | MCLK clock frequency | Off | — | 0 | — | MHz |
| | | Low Frequency (Default) | — | 7.5 | — | MHz |
| | | Medium Frequency[3] | — | 24 | — | MHz |
| | | High Frequency[3] | — | 40 | — | MHz |

## sysCONFIG Port Timing Specifications[1] (Continued)

| Symbol | Parameter | | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $t_{MCLK}$ | CRESET_B high to first MCLK edge | iCE40LP384 - Low Frequency (Default) | 600 | — | — | us |
| | | iCE40LP384 - Medium Frequency | 600 | — | — | us |
| | | iCE40LP384 - High Frequency | 600 | — | — | us |
| | | iCE40LP640, iCE40LP/HX1K - Low Frequency (Default) | 800 | — | — | us |
| | | iCE40LP640, iCE40LP/HX1K - Medium Frequency | 800 | — | — | us |
| | | iCE40LP640, iCE40LP/HX1K - High Frequency | 800 | — | — | us |
| | | iCE40LP/HX1K-Low Frequency (Default) | 800 | — | — | us |
| | | iCE40LP/HX1K - Medium Frequency | 800 | — | — | us |
| | | iCE40LP/HX1K - High Frequency | 800 | — | — | us |
| | | iCE40LP/HX4K - Low Frequency (Default) | 1200 | — | — | us |
| | | iCE40LP/HX4K - Medium Frequency | 1200 | — | — | us |
| | | iCE40LP/HX4K - high frequency | 1200 | — | — | us |
| | | iCE40LP/HX8K - Low Frequency (Default) | 1200 | — | — | us |
| | | iCE40LP/HX8K - Medium Frequency | 1200 | — | — | us |
| | | iCE40LP/HX8K - High Frequency | 1200 | — | — | us |

1. Does not apply for NVCM
2. Supported only with 1.2V $V_{CC}$ and at 25C
3. Extended range $f_{MAX}$ Write operations support up to 53MHz only with 1.2V $V_{CC}$ and at 25C

## Switching Test Conditions

Figure 3-3 shows the output test load used for AC testing. The specific values for resistance, capacitance, voltage, and other test conditions are shown in Table 3-3.

***Figure 3-3. Output Test Load, LVCMOS Standards***

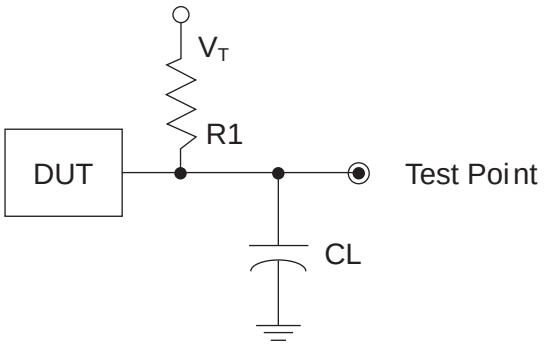



***Table 3-3. Test Fixture Required Components, Non-Terminated Interfaces***

| Test Condition | $R_1$ | $C_L$ | Timing Reference | $V_T$ |
|---|---|---|---|---|
| LVCMOS settings (L -> H, H -> L) | ∞ | 0 pF | LVCMOS 3.3 = 1.5V | — |
| | | | LVCMOS 2.5 = $V_{CCIO}/2$ | — |
| | | | LVCMOS 1.8 = $V_{CCIO}/2$ | — |
| LVCMOS 3.3 (Z -> H) | 188 | 0 pF | 1.5 | $V_{OL}$ |
| LVCMOS 3.3 (Z -> L) | | | 1.5 | $V_{OH}$ |
| Other LVCMOS (Z -> H) | | | $V_{CCIO}/2$ | $V_{OL}$ |
| Other LVCMOS (Z -> L) | | | $V_{CCIO}/2$ | $V_{OH}$ |
| LVCMOS (H -> Z) | | | $V_{OH}$ - 0.15 | $V_{OL}$ |
| LVCMOS (L -> Z) | | | $V_{OL}$ - 0.15 | $V_{OH}$ |

Note: Output test conditions for all other interfaces are determined by the respective standards.

## Signal Descriptions

| Signal Name | I/O | Descriptions |
|---|---|---|
| **General Purpose** | | |
| IO[Bank]_[Row/Column Number][A/B] | I/O | [Bank] indicates the bank of the device on which the pad is located.<br>[Number] indicates IO number on the device. |
| IO[Bank]_[Row/Column Number][A/B] | I/O | [Bank] indicates the bank of the device on which the pad is located.<br>[Number] indicates IO number on the device.<br>[A/B] indicates the differential I/O. 'A' = negative input. 'B' = positive input. |
| HCIO[Bank]_[Number] | I/O | High Current IO. [Bank] indicates the bank of the device on which the pad is located.<br>[Number] indicates IO number. |
| NC | — | No connect |
| GND | — | GND – Ground. Dedicated pins. It is recommended that all GNDs are tied together. |
| VCC | — | VCC – The power supply pins for core logic. Dedicated pins. It is recommended that all VCCs are tied to the same supply. |
| VCCIO_x | — | VCCIO – The power supply pins for I/O Bank x. Dedicated pins. All VCCIOs located in the same bank are tied to the same supply. |
| **PLL and Global Functions (Used as user-programmable I/O pins when not used for PLL or clock pins)** | | |
| VCCPLLx | — | PLL VCC – Power. Dedicated pins. The PLL requires a separate power and ground that is quiet and stable to reduce the output clock jitter of the PLL. |
| GNDPLLx | — | PLL GND – Ground. Dedicated pins. The sysCLOCK PLL has the DC ground connection made on the FPGA, so the external PLL ground connection (GNDPLL) must NOT be connected to the board's ground. |
| GBINx | — | Global pads. Two per side. |
| **Programming and Configuration** | | |
| CBSEL[0:1] | I/O | Dual function pins. I/Os when not used as CBSEL. Optional ColdBoot configuration SELect input, if ColdBoot mode is enabled. |
| CRESET_B | I | Configuration Reset, active Low. Dedicated input. No internal pull-up resistor. Either actively drive externally or connect a 10 KOhm pull-up resistor to VCCIO_2. |
| CDONE | I/O | Configuration Done. Includes a permanent weak pull-up resistor to VCCIO_2. If driving external devices with CDONE output, an external pull-up resistor to VCCIO_2 may be required. Refer to the TN1248, iCE40 Programming and Configuration for more details. Following device configuration the iCE40LP640 and iCE40LP1K in the SWG16 package CDONE pin can be used as a user output. |
| VCC_SPI | — | SPI interface voltage supply input. Must have a valid voltage even if configuring from NVCM. |
| SPI_SCK | I/O | Input Configuration Clock for configuring an FPGA in Slave SPI mode. Output Configuration Clock for configuring an FPGA configuration modes. |
| SPI_SS_B | I/O | SPI Slave Select. Active Low. Includes an internal weak pull-up resistor to VCC_SPI during configuration. During configuration, the logic level sampled on this pin determines the configuration mode used by the iCE40 device. An input when sampled at the start of configuration. An input when in SPI Peripheral configuration mode (SPI_SS_B = Low). An output when in Master SPI Flash configuration mode. |
| SPI_SI | I/O | Slave SPI serial data input and master SPI serial data output |
| SPI_SO | I/O | Slave SPI serial data output and master SPI serial data input |

## Signal Descriptions (Cont.)

| Signal Name | I/O | Descriptions |
|---|---|---|
| VPP_FAST | — | Optional fast NVCM programming supply. $V_{PP\_FAST}$, used only for fast production programming, must be left floating or unconnected in applications, except CM36 and CM49 packages MUST have the $V_{PP\_FAST}$ ball connected to $V_{CCIO\_0}$ ball externally. |
| VPP_2V5 | — | VPP_2V5 NVCM programming and operating supply |

## Pin Information Summary

| | iCE40LP384 | | | iCE40LP640 | iCE40LP1K | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | SG32 | CM36[2] | CM49[2] | SWG16 | SWG16 | CM36[1, 2] | CM49[2] | CM81 | CB81 | QN84 | CM121 | CB121 |
| **General Purpose I/O per Bank** | | | | | | | | | | | | |
| Bank 0 | 6 | 4 | 10 | 3 | 3 | 4 | 10 | 17 | 17 | 17 | 24 | 24 |
| Bank 1 | 5 | 7 | 7 | 0 | 0 | 7 | 7 | 15 | 16 | 17 | 25 | 21 |
| Bank 2 | 0 | 4 | 4 | 1 | 1 | 4 | 4 | 11 | 8 | 11 | 18 | 19 |
| Bank 3 | 6 | 6 | 12 | 2 | 2 | 6 | 10 | 16 | 17 | 18 | 24 | 24 |
| Configuration | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| Total General Purpose Single Ended I/O | 21 | 25 | 37 | 10 | 10 | 25 | 35 | 63 | 62 | 67 | 95 | 92 |
| **High Current Outputs per Bank** | | | | | | | | | | | | |
| Bank 0 | 0 | 0 | 0 | 3 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bank 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bank 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bank 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Total Current Outputs | 0 | 0 | 0 | 3 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Differential Inputs per Bank** | | | | | | | | | | | | |
| Bank 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bank 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bank 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bank 3 | 3 | 3 | 6 | 1 | 1 | 3 | 5 | 8 | 9 | 7 | 12 | 12 |
| Total Differential Inputs | 3 | 3 | 6 | 1 | 1 | 3 | 5 | 8 | 9 | 7 | 12 | 12 |
| **Dedicated Inputs per Bank** | | | | | | | | | | | | |
| Bank 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bank 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bank 2 | 2 | 2 | 2 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| Bank 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Configuration | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Total Dedicated Inputs | 2 | 2 | 2 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| **Vccio Pins** | | | | | | | | | | | | |
| Bank 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 1 |
| Bank 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 2 | 1 |
| Bank 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 1 |
| Bank 3 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 2 | 2 |
| VCC | 1 | 1 | 2 | 1 | 1 | 1 | 2 | 3 | 3 | 4 | 4 | 4 |
| VCC_SPI | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| VPP_2V5 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| VPP_FAST[3] | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| VCCPLL | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| GND | 2 | 3 | 3 | 2 | 2 | 3 | 4 | 5 | 8 | 4 | 8 | 11 |
| NC | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 |
| Total Count of Bonded Pins | 32 | 36 | 49 | 16 | 16 | 36 | 49 | 81 | 81 | 84 | 121 | 121 |

1. $V_{CCIO0}$ and $V_{CCIO1}$ are connected together.
2. $V_{CCIO2}$ and $V_{CCIO3}$ are connected together.
3. $V_{PP\_FAST}$, used only for fast production programming, must be left floating or unconnected in applications, except CM36 and CM49 packages MUST have the $V_{PP\_FAST}$ ball connected to $V_{CCIO\_0}$ ball externally.

## Pin Information Summary, Continued

| | iCE40LP4K | | | iCE40LP8K | | | iCE40HX1K | | | iCE40HX4K | | iCE40HX8K | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | CM81 | CM121 | CM225 | CM81 | CM121 | CM225 | VQ100 | CB132 | TQ144 | CB132 | TQ144 | CB132 | CM225 | CT256 |
| **General Purpose I/O per Bank** | | | | | | | | | | | | | | |
| Bank 0 | 17 | 23 | 46 | 17 | 23 | 46 | 19 | 24 | 23 | 24 | 27 | 24 | 46 | 52 |
| Bank 1 | 15 | 21 | 42 | 15 | 21 | 42 | 19 | 25 | 25 | 25 | 29 | 25 | 42 | 52 |
| Bank 2 | 9 | 19 | 40 | 9 | 19 | 40 | 12 | 20 | 20 | 18 | 19 | 18 | 40 | 46 |
| Bank 3 | 18 | 26 | 46 | 18 | 26 | 46 | 18 | 22 | 24 | 24 | 28 | 24 | 46 | 52 |
| Configuration | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| Total General Purpose Single Ended I/O | 63 | 93 | 178 | 63 | 93 | 178 | 72 | 95 | 96 | 95 | 107 | 95 | 178 | 206 |
| **High Current Outputs per Bank** | | | | | | | | | | | | | | |
| Bank 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bank 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bank 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bank 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Total Differential Inputs | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Differential Inputs per Bank** | | | | | | | | | | | | | | |
| Bank 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bank 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bank 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bank 3 | 9 | 13 | 23 | 9 | 13 | 23 | 9 | 11 | 12 | 12 | 14 | 12 | 23 | 26 |
| Total Differential Inputs | 9 | 13 | 23 | 9 | 13 | 23 | 9 | 11 | 12 | 12 | 14 | 12 | 23 | 26 |
| **Dedicated Inputs per Bank** | | | | | | | | | | | | | | |
| Bank 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bank 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Bank 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| Bank 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Configuration | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Total Dedicated Inputs | 2 | 2 | 3 | 2 | 2 | 3 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| **Vccio Pins** | | | | | | | | | | | | | | |
| Bank 0 | 1 | 1 | 3 | 1 | 1 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 4 |
| Bank 1 | 1 | 1 | 3 | 1 | 1 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 4 |
| Bank 2 | 1 | 1 | 3 | 1 | 1 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 4 |
| Bank 3 | 1 | 2 | 4 | 1 | 2 | 4 | 3 | 3 | 2 | 3 | 2 | 3 | 4 | 4 |
| VCC | 3 | 4 | 8 | 3 | 4 | 8 | 4 | 5 | 4 | 5 | 4 | 5 | 8 | 6 |
| VCC_SPI | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| VPP_2V5 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| VPP_FAST[1] | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| VCCPLL | 1 | 2 | 2 | 1 | 2 | 2 | 0 | 1 | 1 | 2 | 2 | 2 | 2 | 2 |
| GND | 5 | 12 | 18 | 5 | 12 | 18 | 10 | 14 | 10 | 15 | 11 | 15 | 18 | 20 |
| NC | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 19 | 0 | 6 | 0 | 0 | 0 |
| Total Count of Bonded Pins | 81 | 121 | 225 | 81 | 121 | 225 | 100 | 132 | 144 | 132 | 144 | 132 | 225 | 256 |

1. $V_{PP\_FAST}$, used only for fast production programming, must be left floating or unconnected in applications.

# iCE40 LP/HX Family Data Sheet Ordering Information

October 2013 Data Sheet DS1040

## iCE40 Part Number Description

### Ultra Low Power (LP) Devices

iCE40LPXXX – XXXXX

**Device Family**
iCE40 FPGA

**Series**
LP = Low Power Series

**Logic Cells**
384 = 384 Logic Cells
640 = 640 Logic Cells
1K = 1,280 Logic Cells
4K = 3,520 Logic Cells
8K = 7,680 Logic Cells

**Package**
SWG16TR = 16-Ball WLCSP (0.35 mm Pitch) in tape and reel
SWG16TR50 = 16-Ball WLCSP (0.35 mm Pitch) in tape and reel 50 units
SWG16TR1K = 16-Ball WLCSP (0.35 mm Pitch) in tape and reel 1,000 units
CM36 = 36-Ball ucBGA (0.4 mm Pitch)
CM36TR = 36-Ball ucBGA (0.4 mm Pitch) in tape and reel
CM49 = 49-Ball ucBGA (0.4 mm Pitch)
CM49TR = 49-Ball ucBGA (0.4 mm Pitch) in tape and reel
CM81 = 81-Ball ucBGA (0.4 mm Pitch)
CM81TR = 81-Ball ucBGA (0.4 mm Pitch) in tape and reel
CB81 = 81-Ball csBGA (0.5 mm Pitch)
CM121 = 121-Ball ucBGA (0.4 mm Pitch)
CB121 = 121-Ball csBGA (0.5 mm Pitch)
CM225 = 225-Ball ucBGA (0.4 mm Pitch)
SG32 = 32-Pin QFN (0.5 mm Pitch)
QN84 = 84-Pin QFN (0.5 mm Pitch)

### High Performance (HX) Devices

iCE40HXXX – XXXXX

**Device Family**
iCE40 mobileFPGA

**Series**
HX = High-Performance Series

**Logic Cells**
1K = 1,280 Logic Cells
4K = 3,520 Logic Cells
8K = 7,680 Logic Cells

**Package**
CB132 = 132-Ball csBGA (0.5 mm Pitch)
CM225 = 225-Ball ucBGA (0.4 mm Pitch)
CT256 = 256-Ball caBGA (0.8 mm Pitch)
TQ144 = 144-Pin TQFP (0.5 mm Pitch)
VQ100 = 100-Pin VQFP (0.5 mm Pitch)

All parts shipped in trays unless noted.

## Ordering Information

iCE40 devices have top-side markings as shown below:

**Industrial**


Note: Markings are abbreviated for small packages.

## Ultra Low Power Industrial Grade Devices, Halogen Free (RoHS) Packaging

| Part Number | LUTs | Supply Voltage | Package | Leads | Temp. |
|---|---|---|---|---|---|
| iCE40LP384-CM36 | 384 | 1.2V | Halogen-Free ucBGA | 36 | IND |
| iCE40LP384-CM36TR | 384 | 1.2V | Halogen-Free ucBGA | 36 | IND |
| iCE40LP384-CM49 | 384 | 1.2V | Halogen-Free ucBGA | 49 | IND |
| iCE40LP384-CM49TR | 384 | 1.2V | Halogen-Free ucBGA | 49 | IND |
| iCE40LP384-SG32 | 384 | 1.2V | Halogen-Free QFN | 32 | IND |
| iCE40LP640-SWG16TR | 640 | 1.2V | Halogen-Free WLCSP | 16 | IND |
| iCE40LP640-SWG16TR50 | 640 | 1.2V | Halogen-Free WLCSP | 16 | IND |
| iCE40LP640-SWG16TR1K | 640 | 1.2V | Halogen-Free WLCSP | 16 | IND |
| iCE40LP1K-SWG16TR | 1280 | 1.2V | Halogen-Free WLCSP | 16 | IND |
| iCE40LP1K-SWG16TR50 | 1280 | 1.2V | Halogen-Free WLCSP | 16 | IND |
| iCE40LP1K-SWG16TR1K | 1280 | 1.2V | Halogen-Free WLCSP | 16 | IND |
| iCE40LP1K-CM36 | 1280 | 1.2V | Halogen-Free ucBGA | 36 | IND |
| iCE40LP1K-CM36TR | 1280 | 1.2V | Halogen-Free ucBGA | 36 | IND |
| iCE40LP1K-CM49 | 1280 | 1.2V | Halogen-Free ucBGA | 49 | IND |
| iCE40LP1K-CM49TR | 1280 | 1.2V | Halogen-Free ucBGA | 49 | IND |
| iCE40LP1K-CM81 | 1280 | 1.2V | Halogen-Free ucBGA | 81 | IND |
| iCE40LP1K-CM81TR | 1280 | 1.2V | Halogen-Free ucBGA | 81 | IND |
| iCE40LP1K-CB81 | 1280 | 1.2V | Halogen-Free csBGA | 81 | IND |
| iCE40LP1K-CM121 | 1280 | 1.2V | Halogen-Free ucBGA | 121 | IND |
| iCE40LP1K-CB121 | 1280 | 1.2V | Halogen-Free csBGA | 121 | IND |
| iCE40LP1K-QN84 | 1280 | 1.2V | Halogen-Free QFN | 84 | IND |
| iCE40LP4K-CM81 | 3520 | 1.2V | Halogen-Free ucBGA | 81 | IND |
| iCE40LP4K-CM81TR | 3520 | 1.2V | Halogen-Free ucBGA | 81 | IND |
| iCE40LP4K-CM121 | 3520 | 1.2V | Halogen-Free ucBGA | 121 | IND |
| iCE40LP4K-CM225 | 3520 | 1.2V | Halogen-Free ucBGA | 225 | IND |
| iCE40LP8K-CM81 | 7680 | 1.2V | Halogen-Free ucBGA | 81 | IND |
| iCE40LP8K-CM121 | 7680 | 1.2V | Halogen-Free ucBGA | 121 | IND |
| iCE40LP8K-CM225 | 7680 | 1.2V | Halogen-Free ucBGA | 225 | IND |

## High-Performance Industrial Grade Devices, Halogen Free (RoHS) Packaging

| Part Number | LUTs | Supply Voltage | Package | Leads | Temp. |
|---|---|---|---|---|---|
| iCE40HX1K-CB132 | 1280 | 1.2V | Halogen-Free csBGA | 132 | IND |
| iCE40HX1K-VQ100 | 1280 | 1.2V | Halogen-Free VQFP | 100 | IND |
| iCE40HX1K-TQ144 | 1280 | 1.2V | Halogen-Free TQFP | 144 | IND |
| iCE40HX4K-CB132 | 3520 | 1.2V | Halogen-Free csBGA | 132 | IND |
| iCE40HX4K-TQ144 | 3520 | 1.2V | Halogen-Free TQFP | 144 | IND |
| iCE40HX8K-CB132 | 7680 | 1.2V | Halogen-Free csBGA | 132 | IND |
| iCE40HX8K-CM225 | 7680 | 1.2V | Halogen-Free ucBGA | 225 | IND |
| iCE40HX8K-CT256 | 7680 | 1.2V | Halogen-Free caBGA | 256 | IND |

# iCE40 LP/HX Family Data Sheet Supplemental Information

**March 2013** **Data Sheet DS1040**

## For Further Information

A variety of technical notes for the iCE40 family are available on the Lattice web site.

- TN1248, iCE40 Programming and Configuration
- TN1250, Memory Usage Guide for iCE40 Devices
- TN1251, iCE40 sysCLOCK PLL Design and Usage Guide
- TN1252, iCE40 Hardware Checklist
- TN1253, Using Differential I/O (LVDS, Sub-LVDS) in iCE40 Devices
- TN1074, PCB Layout Recommendations for BGA Packages
- iCE40 Pinout Files
- Thermal Management document
- Lattice design tools
- IBIS
- Package Data Sheet
- Schematic Symbols

# iCE40 LP/HX Family Data Sheet Revision History

**October 2013** **Data Sheet DS1040**

| Date | Version | Section | Change Summary |
|---|---|---|---|
| July 2011 | 01.0 | — | Initial release. |
| July 2011 | 01.01 | — | Added 640, 1K and 4K to Table 13 configuration times. Updated Table 1 maximum I/Os. |
| July 2011 | 01.1 | — | Moved package specifications to iCE40 pinout Excel files. |
| | | | Updated Table 1 maximum I/Os. |
| Aug 2012 | 01.2 | — | Updated company name. |
| July 2011 | 01.21 | — | Updated Figure 3 and Figure 4 to specify iCE40. |
| July 2011 | 01.3 | — | Production release. |
| | | | Updated notes on Table 3: Recommended Operating Conditions. |
| | | | Updated values in Table 4, Table 5, Table 12, Table 13 and Table 17. |
| July 2011 | 01.31 | — | Updated Table 1. |
| | 02.0 | — | Merged SiliconBlue iCE40 LP and HX data sheets and updated to Lattice format. |
| March 2013 | 02.1 | DC and Switching Characteristics | Recommended operating conditions added requirement for Master SPI. |
| | | | Updated Recommended Operating Conditions for $V_{PP\_2V5}$. |
| | | | Updated Power-On-Reset Voltage Levels and sequence requirements. |
| | | | Updated Static Supply Current conditions. |
| | | | Changed unit for $t_{SKEW\_IO}$ from ns to ps. |
| | | | Updated range of CCLK $f_{MAX}$. |
| | | Ordering Information | Updated ordering information to include tape and reel part numbers. |
| April 2013 | 02.2 | Introduction | Added the LP8K 81 ucBGA. |
| | | Architecture | Corrected typos. |
| | | DC and Switching Characteristics | Corrected typos.<br>Added 7:1 LVDS waveforms. |
| | | Pinout Information | Corrected typos in signal descriptions.<br>Added the LP8K 81 ucBGA. |
| | | Ordering Information | Added the LP8K 81 ucBGA. |
| May 2013 | 02.3 | DC and Switching Characteristics | Added new data from Characterization. |
| July 2013 | 02.4 | Introduction | Updated the iCE40 Family Selection Guide table. |
| | | DC and Switching Characteristics | Updated the sysCONFIG Port Timing Specifications table. |
| | | | Updated footnote in DC Electrical Characteristics table. |
| | | | GDDR tables removed. Support to be provided in a technical note. |
| | | Pinout Information | Updated the Pin Information Summary table. |
| | | Ordering Information | Updated the top-side markings figure. |
| | | | Updated the Ultra Low Power Industrial Grade Devices, Halogen Free (RoHS) Packaging table. |

| Date | Version | Section | Change Summary |
|---|---|---|---|
| August 2013 | 02.5 | Introduction | Updated the iCE40 Family Selection Guide table. |
| | | DC and Switching Characteristics | Updated the following tables:<br>Absolute Maximum Ratings<br>Power-On-Reset Voltage Levels<br>Static Supply Current – LP Devices<br>Static Supply Current – HX Devices<br>Programming NVCM Supply Current – LP Devices<br>Programming NVCM Supply Current – HX Devices<br>Peak Startup Supply Current – LP Devices<br>sysIO Recommended Operating Conditions<br>Typical Building Block Function Performance – HX Devices<br>iCE40 External Switching Characteristics – HX Devices<br>sysCLOCK PLL Timing – Preliminary<br>SPI Master or NVCM Configuration Time |
| | | Pinout Information | Updated the Pin Information Summary table. |
| September 2013 | 02.6 | DC and Switching Characteristics | Updated Absolute Maximum Ratings section. |
| | | | Updated sysCLOCK PLL Timing – Preliminary table. |
| | | Pinout Information | Updated Pin Information Summary table. |
| October 2013 | 02.7 | Introduction | Updated Features list and iCE40 Family Selection Guide table. |
| | | Architecture | Revised iCE40-1K device to iCE40LP/HX1K device. |
| | | DC and Switching Characteristics | Added iCE40LP640 device information. |
| | | Pinout Information | Added iCE40LP640 and iCE40LP1K information |
| | | Ordering Information | Added iCE40LP640 and iCE40LP1K information |

# Section II. iCE40LM Family Data Sheet

# iCE40LM Family Data Sheet Introduction

October 2013 Advance Data Sheet DS1045

## General Description

iCE40LM family is an ultra-low power FPGA and sensor manager designed for ultra-low power mobile applications, such as smartphones, tablets and hand-held devices. The iCE40LM family includes integrated SPI & $I^2C$ blocks to interface with virtually all mobile sensors and application processors. The iCE40LM family also features two Strobe Generators that can generates strobes in Microsecond ranges with the Low-Power Strobe Generator, and also generates strobes in Nanosecond ranges with the High-Speed Strobe Generator.

In addition, the iCE40LM family of devices includes logic to perform other functions such as mobile bridging, antenna tuning, GPIO expansion, motion/gesture recognition, IR remote control, bar code emulation and other custom functions.

The iCE40LM family features three device densities, from 1000 to 3520 Look Up Tables (LUTs) of logic with programmable I/Os that can be used as either SPI/$I^2C$ interface ports or general purpose I/O's. It also has up to 80Kbits of Block RAMs to work with user logic.

## Features

- **Flexible Logic Architecture**
  - Three devices with 1000 to 3520 LUTs
  - 18 I/O pins for 25-pin WLCSP
- **Ultra-low Power Devices**
  - Advanced 40 nm ultra-low power process
  - As low as 120 µW standby power typical
- **Embedded and Distributed Memory**
  - Up to 80 Kbits sysMEM™ Embedded Block RAM
- **Two Hardened $I^2C$ Interfaces**
- **Two Hardened SPI Interfaces**
- **Two On-Chip Strobe Generators**
  - Low-Power Strobe Generator (Microsecond ranges)
  - High-Speed Strobe Generator (Nanosecond ranges)
- **High Current Drive Outputs for LED**
  - 3 High Drive (HD) output in each device
  - Source/sink nominal 24mA
- **Flexible On-Chip Clocking**
  - Six low-skew global signal resource
- **Flexible Device Configuration**
  - SRAM is configured through SPI
- **Ultra-Small Form Factor**
  - As small as 25-pin WLCSP package 1.71mm x 1.71 mm

## Applications

- Smartphones
- Tablets and Consumer Handheld Devices
- Handheld Commercial and Industrial Devices
- Multi Sensor Management Applications
- Sensor Pre-processing & Sensor Fusion
- Always-On Sensor Applications

*Table 8-1. iCE40LM Family Selection Guide*

| Part Number | iCE40LM1K | iCE40LM2K | iCE40LM4K |
|---|---|---|---|
| Logic Cells (LUT + Flip-Flop) | 1000 | 2000 | 3520 |
| RAM4K Memory Blocks | 16 | 20 | 20 |
| RAM4K RAM Bits | 64K | 80K | 80K |
| Package | Programmable I/O Count | | |
| 25-pin WLCSP, 1.71 x 1.71 mm, 0.35mm | 18 | 18 | 18 |
| 36-pin ucBGA, 2.5 x 2.5 mm, 0.40mm | 28 | 28 | 28 |
| 49-pin ucBGA, 3 x 3 mm, 0.40mm | 37 | 37 | 37 |

## Introduction

The iCE40LM family of ultra-low power FPGAs has three devices with densities ranging from 1000 to 3520 Look-Up Tables (LUTs). In addition to LUT-based, low-cost programmable logic, these devices also feature Embedded Block RAM (EBR), two Strobe Generators (LPSG, HSSG), two hardened $I^2C$ Controllers and two hardened SPI Controllers. These features allow the devices to be used in low-cost, high-volume consumer and mobile applications,

The iCE40LM devices are fabricated on a 40nm CMOS low power process. The device architecture has several features such as user configurable $I^2C$ and SPI Controllers, either as master or slave, and two Strobe Generators.

The iCE40LM FPGAs are available in very small form factor packages, with the smallest in 25-pin WLCSP. The 25-pin WLCSP package has a 0.35mm ball pitch, resulting to an overall package size of 1.71mm x 1.71mm that easily fits into a lot of mobile applications. Table 8-1 shows the LUT densities, package and I/O pin count.

The iCE40LM devices offer enhanced I/O features such as pull-up resistors. Pull-up features are controllable on a “per-pin” basis.

Lattice provides a variety of design tools that allow complex designs to be efficiently implemented using the iCE40LM family of devices. Popular logic synthesis tools provide synthesis library support for iCE40LM. Lattice design tools use the synthesis tool output along with the user-specified preferences and constraints to place and route the design in the iCE40LM device. These tools extract the timing from the routing and back-annotate it into the design for timing verification.

Lattice provides many pre-engineered IP (Intellectual Property) modules, including a number of reference designs, licensed free of charge, optimized for the iCE40LM FPGA family. Lattice also can provide fully verified bit-stream for some of the widely used target functions in mobile device applications, such as ultra-low power sensor management, gesture recognition, IR remote, barcode emulator functions. Users can use these functions as offered by Lattice, or they can use the design to create their own unique required functions. For more information regarding Lattice's Reference Designs or fully-verified bitstreams, please contact your local Lattice representative.

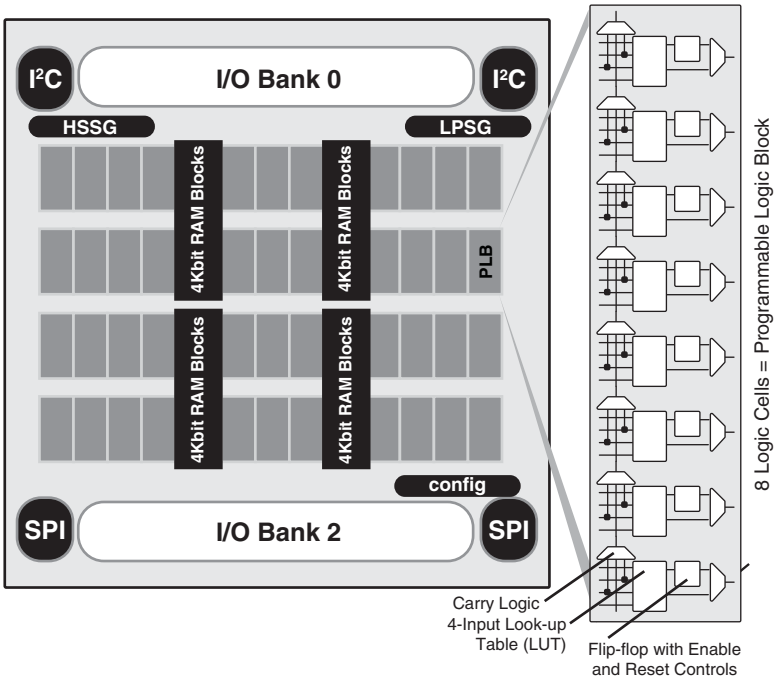
# iCE40LM Family Data Sheet Architecture



## Architecture Overview

The iCE40LM family architecture contains an array of Programmable Logic Blocks (PLB), two Strobe Generators, two user configurable I²C controllers, two user configurable SPI controllers, and blocks of sysMEM™ Embedded Block RAM (EBR) surrounded by Programmable I/O (PIO). Figure 9-1shows the block diagram of the iCE40LM-4K device.

***Figure 9-1. iCE40LM-4K Device, Top View***




The logic blocks, Programmable Logic Blocks (PLB) and sysMEM EBR blocks, are arranged in a two-dimensional grid with rows and columns. Each column has either logic blocks or EBR blocks. The PIO cells are located at the top and bottom of the device, arranged in banks. The PLB contains the building blocks for logic, arithmetic, and register functions. The PIOs utilize a flexible I/O buffer referred to as a sysIO buffer that supports operation with a variety of interface standards. The blocks are connected with many vertical and horizontal routing channel resources. The place and route software tool automatically allocates these routing resources.

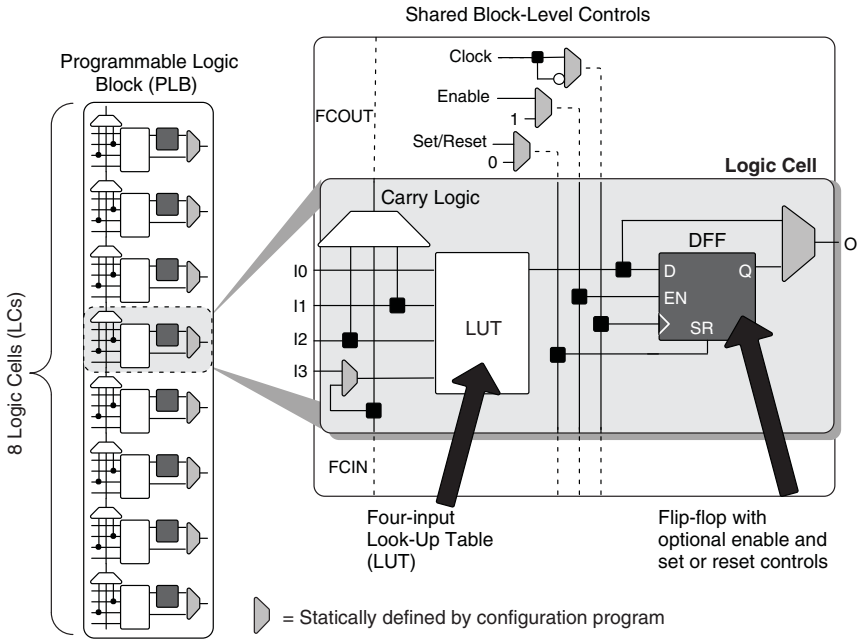
In the iCE40LM family, There are two sysIO banks, one on top and one on bottom. User can connect both $V_{CCIO}$s together, if all the I/Os are using the same voltage standard. Refer to the details in later sections of this document. The sysMEM EBRs are large 4 Kbit, dedicated fast memory blocks. These blocks can be configured as RAM, ROM or FIFO with user logic using PLBs.

Every device in the family has two user SPI ports, one of these (right side) SPI port also supports programming and configuration of the device. The iCE40LM also includes two user I²C ports, and two Strobe Generators.

## PLB Blocks

The core of the iCE40LM device consists of Programmable Logic Blocks (PLB) which can be programmed to perform logic and arithmetic functions. Each PLB consists of eight interconnected Logic Cells (LC) as shown in Figure 9-2. Each LC contains one LUT and one register.

***Figure 9-2. PLB Block Diagram***




### Logic Cells

Each Logic Cell includes three primary logic elements shown in Figure 9-2.

- A four-input Look-Up Table (LUT) builds any combinational logic function, of any complexity, requiring up to four inputs. Similarly, the LUT element behaves as a 16x1 Read-Only Memory (ROM). Combine and cascade multiple LUTs to create wider logic functions.
- A 'D'-style Flip-Flop (DFF), with an optional clock-enable and reset control input, builds sequential logic functions. Each DFF also connects to a global reset signal that is automatically asserted immediately following device configuration.
- Carry Logic boosts the logic efficiency and performance of arithmetic functions, including adders, subtracters, comparators, binary counters and some wide, cascaded logic functions.

***Table 9-1. Logic Cell Signal Descriptions***

| Function | Type | Signal Names | Description |
|---|---|---|---|
| Input | Data signal | I0, I1, I2, I3 | Inputs to LUT |
| Input | Control signal | Enable | Clock enable shared by all LCs in the PLB |
| Input | Control signal | Set/Reset[1] | Asynchronous or synchronous local set/reset shared by all LCs in the PLB. |
| Input | Control signal | Clock | Clock one of the eight Global Buffers, or from the general-purpose interconnects fabric shared by all LCs in the PLB |
| Input | Inter-PLB signal | FCIN | Fast carry in |
| Output | Data signals | O | LUT or registered output |
| Output | Inter-PFU signal | FCOUT | Fast carry out |

1. If Set/Reset is not used, then the flip-flop is never set/reset, except when cleared immediately after configuration.

## Routing

There are many resources provided in the iCE40LM devices to route signals individually with related control signals. The routing resources consist of switching circuitry, buffers and metal interconnect (routing) segments.

The inter-PLB connections are made with three different types of routing resources: Adjacent (spans two PLBs), x4 (spans five PLBs) and x12 (spans thirteen PLBs). The Adjacent, x4 and x12 connections provide fast and efficient connections in the diagonal, horizontal and vertical directions.

The design tool takes the output of the synthesis tool and places and routes the design.

## Clock/Control Distribution Network

Each iCE40LM device has six global inputs, two pins on the top bank and four pins on the bottom bank

These global inputs can be used as high fanout nets, clock, reset or enable signals. The dedicated global pins are identified as Gxx and the global buffers are identified as GBUF[7:0]. These six inputs may be used as general purpose I/O if they are not used to drive the clock nets.

Table 9-2 lists the connections between a specific global buffer and the inputs on a PLB. All global buffers optionally connect to the PLB CLK input. Any four of the eight global buffers can drive logic inputs to a PLB. Even-numbered global buffers optionally drive the Set/Reset input to a PLB. Similarly, odd-numbered buffers optionally drive the PLB clock-enable input. GBUF[7:6, 3:0] can connect directly to G[7:6, 3:0] pins respectively. GBUF4 and GBUF5 can connect to the two on-chip Strobe Generators (GBUF4 connects to LPSG, GBUF5 connects to HSSG).

***Table 9-2. Global Buffer (GBUF) Connections to Programmable Logic Blocks***

| Global Buffer | LUT Inputs | Clock | Clock Enable | Reset |
|---|---|---|---|---|
| GBUF0 | Yes, any 4 of 8 GBUF Inputs | ✓ | ✓ | |
| GBUF1 | | ✓ | | ✓ |
| GBUF2 | | ✓ | ✓ | |
| GBUF3 | | ✓ | | ✓ |
| GBUF4 | | ✓ | ✓ | |
| GBUF5 | | ✓ | | ✓ |
| GBUF6 | | ✓ | ✓ | |
| GBUF7 | | ✓ | | ✓ |

The maximum frequency for the global buffers are shown in the iCE40LM External Switching Characteristics tables later in this document.

### Global Hi-Z Control

The global high-impedance control signal, GHIZ, connects to all I/O pins on the iCE40LM device. This GHIZ signal is automatically asserted throughout the configuration process, forcing all user I/O pins into their high-impedance state.

### Global Reset Control

The global reset control signal connects to all PLB and PIO flip-flops on the iCE40LM device. The global reset signal is automatically asserted throughout the configuration process, forcing all flip-flops to their defined wake-up state. For PLB flip-flops, the wake-up state is always reset, regardless of the PLB flip-flop primitive used in the application.

## sysCLOCK Phase Locked Loops (PLLs) - *NOT SUPPORTED on the 25-Pin WLCSP*

The sysCLOCK PLLs provide the ability to synthesize clock frequencies. The iCE40LM devices have one sysCLOCK PLL (Please note that the 25-pin WLCSP package does not support the PLL). REFERENCECLK is the reference frequency input to the PLL and its source can come from an external I/O pin, the internal strobe generator or from internal routing. EXTFEEDBACK is the feedback signal to the PLL which can come from internal routing or an external I/O pin. The feedback divider is used to multiply the reference frequency and thus synthesize a higher frequency clock output.

The PLLOUT output has an output divider, thus allowing the PLL to generate different frequencies for each output. The output divider can have a value from 1 to 64 (in increments of 2X). The PLLOUT outputs can all be used to drive the iCE40 global clock network directly or general purpose routing resources can be used.

The LOCK signal is asserted when the PLL determines it has achieved lock and de-asserted if a loss of lock is detected. A block diagram of the PLL is shown in Figure 9-3.

The timing of the device registers can be optimized by programming a phase shift into the PLLOUT output clock which will advance or delay the output clock with reference to the REFERENCECLK clock. This phase shift can be either programmed during configuration or can be adjusted dynamically. In dynamic mode, the PLL may lose lock after a phase adjustment on the output used as the feedback source and not relock until the tLOCK parameter has been satisfied.

The iCE40LM PLL functions the same as the PLLs in the iCE40 family. For more details on the PLL, see TN1251, iCE40 sysCLOCK PLL Design and Usage Guide.

***Figure 9-3. PLL Diagram***

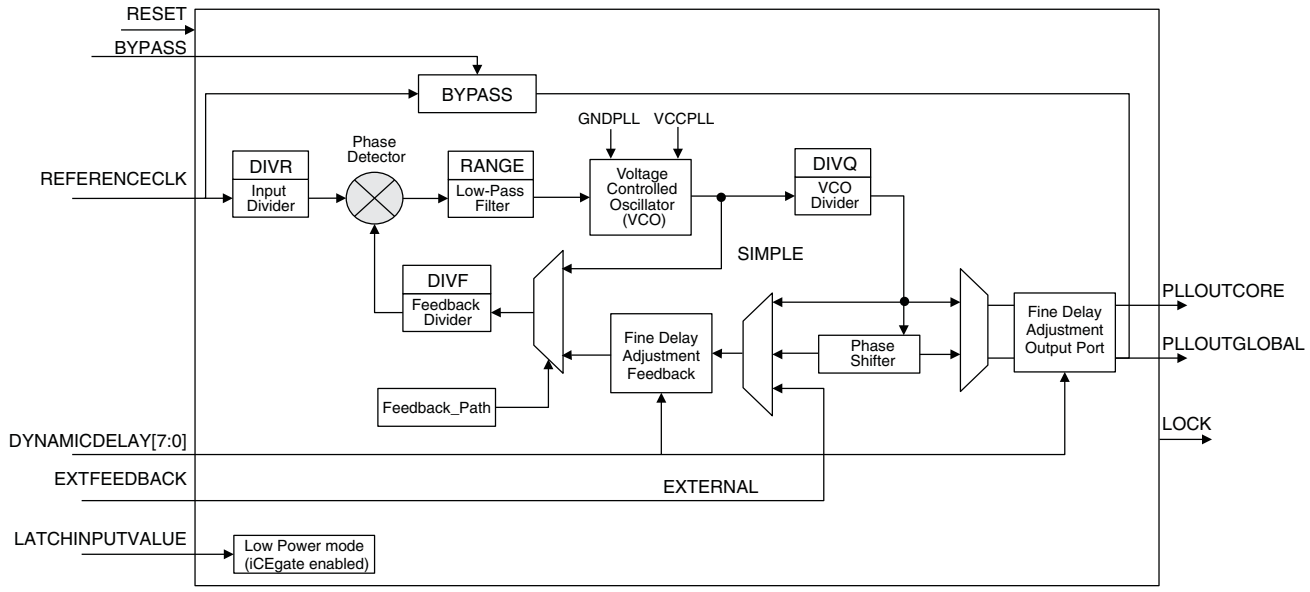



Table 9-3 provides signal descriptions of the PLL block.

*Table 9-3. PLL Signal Descriptions*

| Signal Name | Direction | Description |
|---|---|---|
| REFERENCECLK | Input | Input reference clock |
| BYPASS | Input | The BYPASS control selects which clock signal connects to the PLL-OUT output.<br>0 = PLL generated signal<br>1 = REFERENCECLK |
| EXTFEEDBACK | Input | External feedback input to PLL. Enabled when the FEEDBACK_PATH attribute is set to EXTERNAL. |
| DYNAMICDELAY[7:0] | Input | Fine delay adjustment control inputs. Enabled when DELAY_ADJUSTMENT_MODE is set to DYNAMIC. |
| LATCHINPUTVALUE | Input | When enabled, forces the PLL into low-power mode; PLL output is held static at the last input clock value. Set ENABLE ICEGATE_PORTA and PORTB to '1' to enable. |
| PLLOUTGLOBAL | Output | Output from the Phase-Locked Loop (PLL). Drives a global clock network on the FPGA. The port has optimal connections to global clock buffers GBUF4 and GBUF5. |
| PLLOUTCORE | Output | Output clock generated by the PLL, drives regular FPGA routing. The frequency generated on this output is the same as the frequency of the clock signal generated on the PLLOUTLGOBAL port. |
| LOCK | Output | When High, indicates that the PLL output is phase aligned or locked to the input reference clock. |
| RESET | Input | Active low reset. |

## sysMEM Embedded Block RAM Memory

Larger iCE40LM device includes multiple high-speed synchronous sysMEM Embedded Block RAMs (EBRs), each 4 Kbit in size. This memory can be used for a wide variety of purposes including data buffering, and FIFO.

### sysMEM Memory Block

The sysMEM block can implement single port, pseudo dual port, or FIFO memories with programmable logic resources. Each block can be used in a variety of depths and widths as shown in Table 9-4.

*Table 9-4. sysMEM Block Configurations[1]*

| Block RAM Configuration | Block RAM Configuration and Size | WADDR Port Size (Bits) | WDATA Port Size (Bits) | RADDR Port Size (Bits) | RDATA Port Size (Bits) | MASK Port Size (Bits) |
|---|---|---|---|---|---|---|
| SB_RAM256x16<br>SB_RAM256x16NR<br>SB_RAM256x16NW<br>SB_RAM256x16NRNW | 256x16 (4K) | 8 [7:0] | 16 [15:0] | 8 [7:0] | 16 [15:0] | 16 [15:0] |
| SB_RAM512x8<br>SB_RAM512x8NR<br>SB_RAM512x8NW<br>SB_RAM512x8NRNW | 512x8 (4K) | 9 [8:0] | 8 [7:0] | 9 [8:0] | 8 [7:0] | No Mask Port |
| SB_RAM1024x4<br>SB_RAM1024x4NR<br>SB_RAM1024x4NW<br>SB_RAM1024x4NRNW | 1024x4 (4K) | 10 [9:0] | 4 [3:0] | 10 [9:0] | 4 [3:0] | No Mask Port |
| SB_RAM2048x2<br>SB_RAM2048x2NR<br>SB_RAM2048x2NW<br>SB_RAM2048x2NRNW | 2048x2 (4K) | 11 [10:0] | 2 [1:0] | 11 [10:0] | 2 [1:0] | No Mask Port |

1. For iCE40LM EBR primitives with a negative-edged Read or Write clock, the base primitive name is appended with a 'N' and a 'R' or 'W' depending on the clock that is affected.

### RAM Initialization and ROM Operation

If desired, the contents of the RAM can be pre-loaded during device configuration.

By preloading the RAM block during the chip configuration cycle and disabling the write controls, the sysMEM block can also be utilized as a ROM.

### Memory Cascading

Larger and deeper blocks of RAM can be created using multiple EBR sysMEM Blocks.

### RAM4k Block

Figure 9-4 shows the 256x16 memory configurations and their input/output names. In all the sysMEM RAM modes, the input data and addresses for the ports are registered at the input of the memory array.

***Figure 9-4. sysMEM Memory Primitives***

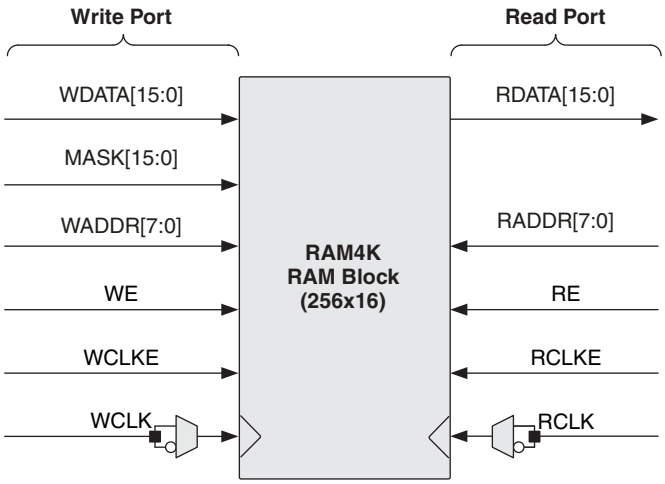



***Table 9-5. EBR Signal Descriptions***

| Signal Name | Direction | Description |
|---|---|---|
| WDATA[15:0] | Input | Write Data input. |
| MASK[15:0] | Input | Masks write operations for individual data bit-lines.<br>0 = write bit; 1 = don't write bit |
| WADDR[7:0] | Input | Write Address input. Selects one of 256 possible RAM locations. |
| WE | Input | Write Enable input. |
| WCLKE | Input | Write Clock Enable input. |
| WCLK | Input | Write Clock input. Default rising-edge, but with falling-edge option. |
| RDATA[15:0] | Output | Read Data output. |
| RADDR[7:0] | Input | Read Address input. Selects one of 256 possible RAM locations. |
| RE | Input | Read Enable input. |
| RCLKE | Input | Read Clock Enable input. |
| RCLK | Input | Read Clock input. Default rising-edge, but with falling-edge option. |

The iCE40LM EBR block functions the same as EBR blocks in the iCE40 family. For further information on the sysMEM EBR block, please refer to TN1250, Memory Usage Guide for iCE40 Devices.

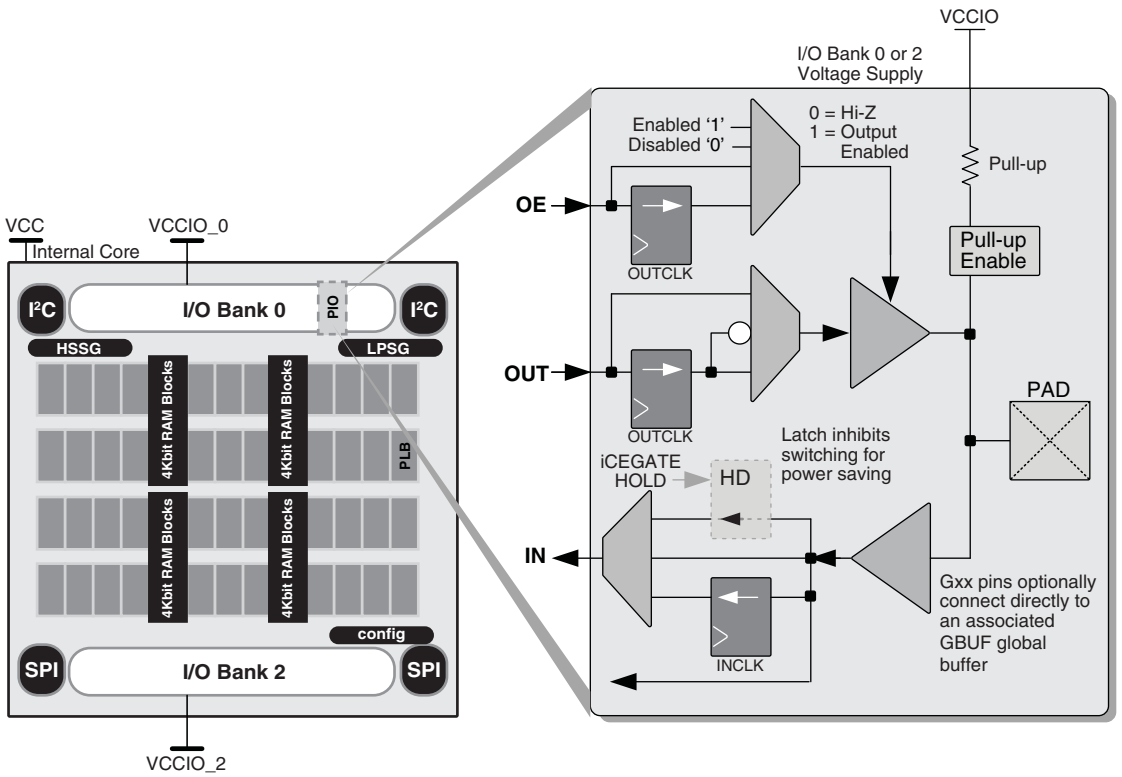
## sysIO Buffer Banks

iCE40LM devices have up to two I/O banks with independent $V_{CCIO}$ rails. Configuration bank $V_{CC\_SPI}$ for the SPI I/Os is connected to $V_{CCIO2}$ on the 25-pin WLCSP package.

### Programmable I/O (PIO)

The programmable logic associated with an I/O is called a PIO. The individual PIOs are connected to their respective sysIO buffers and pads. The PIOs are placed on the top and bottom of the devices.

***Figure 9-5. I/O Bank and Programmable I/O Cell***




The PIO contains three blocks: an input register block, output register block iCEGate™ and tri-state register block. To save power, the optional iCEGate™ latch can selectively freeze the state of individual, non-registered inputs within an I/O bank. Note that the freeze signal is common to the bank. These blocks can operate in a variety of modes along with the necessary clock and selection logic.

### Input Register Block

The input register blocks for the PIOs on all edges contain registers that can be used to condition high-speed interface signals before they are passed to the device core.

### Output Register Block

The output register block can optionally register signals from the core of the device before they are passed to the sysIO buffers.

Figure 9-6 shows the input/output register block for the PIOs.

*Figure 9-6. iCE I/O Register Block Diagram*

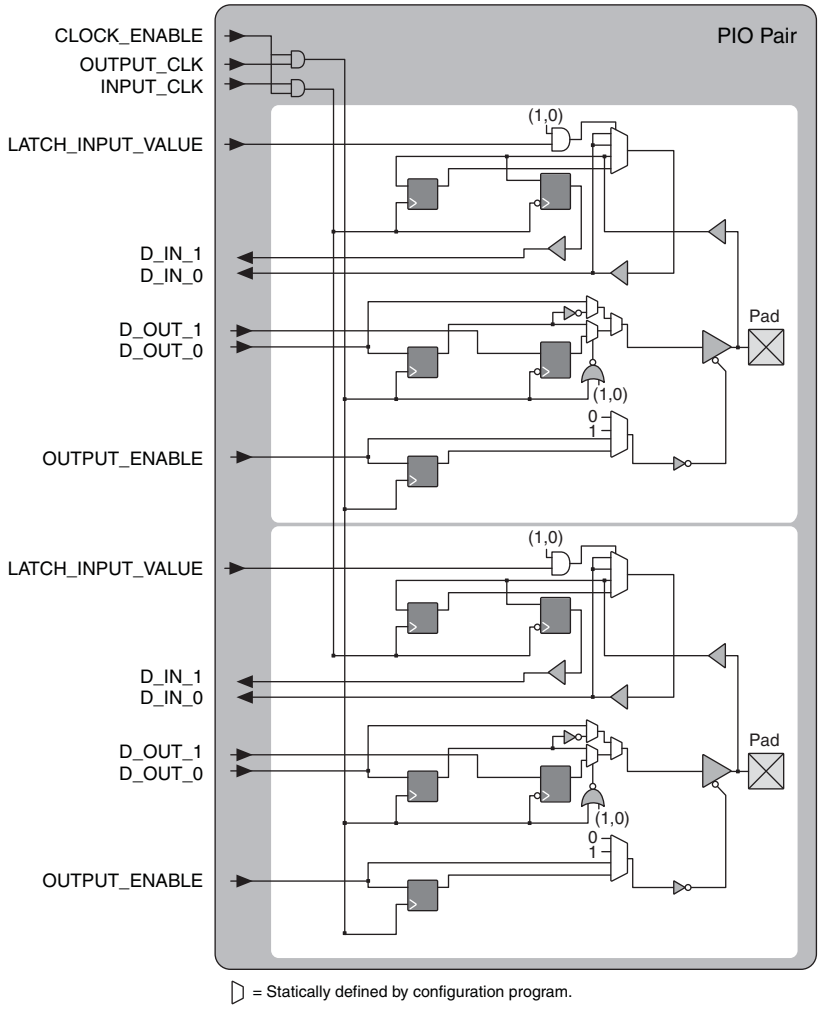



*Table 9-6. PIO Signal List*

| Pin Name | I/O Type | Description |
|---|---|---|
| OUTPUT_CLK | Input | Output register clock |
| CLOCK_ENABLE | Input | Clock enable |
| INPUT_CLK | Input | Input register clock |
| OUTPUT_ENABLE | Input | Output enable |
| D_OUT_0/1 | Input | Data from the core |
| D_IN_0/1 | Output | Data to the core |
| LATCH_INPUT_VALUE | Input | Latches/holds the Input Value |

## sysIO Buffer

Each I/O is associated with a flexible buffer referred to as a sysIO buffer. These buffers are arranged around the periphery of the device in groups referred to as banks. The sysIO buffers allow users to implement a wide variety of standards that are found in today's systems with LVCMOS interfaces.

### Typical I/O Behavior During Power-up

The internal power-on-reset (POR) signal is deactivated when $V_{CC}$, $V_{CCIO\_2}$ and $V_{CC\_SPI}$ ($V_{CC\_SPI}$ is connected to $V_{CCIO\_2}$ on the 25-pin WLCSP) reach the level defined in the Power-On-Reset Voltage table in the DC and Switching Characteristics section of this data sheet. After the POR signal is deactivated, the FPGA core logic becomes active. You must ensure that all $V_{CCIO}$ banks are active with valid input logic levels to properly control the output logic states of all the I/O banks that are critical to the application. The default configuration of the I/O pins in a device prior to configuration is tri-stated with a weak pull-up to $V_{CCIO}$. The I/O pins maintain the pre-configuration state until $V_{CC}$ and $V_{CCIO\_2}$ reach the defined levels. The I/Os take on the software user-configured settings only after $V_{CC\_SPI}$ reaches the level and the device performs a proper download/configuration. Unused I/Os are automatically blocked and the pull-up termination is disabled.

### Supported Standards

The iCE40LM sysIO buffer supports all single-ended input and output standards. The buffer supports the LVCMOS 1.8, 2.5, and 3.3V standards. The buffer has individually configurable options for bus maintenance (weak pull-up or none).

Table 9-7 and Table 9-8 show the I/O standards (together with their supply and reference voltages) supported by the iCE40LM devices.

***Table 9-7. Supported Input Standards***

| Input Standard | $V_{CCIO}$ (Typical) | | |
|---|---|---|---|
| | 3.3V | 2.5V | 1.8V |
| **Single-Ended Interfaces** | | | |
| LVCMOS33 | ✓ | | |
| LVCMOS25 | | ✓ | |
| LVCMOS18 | | | ✓ |

***Table 9-8. Supported Output Standards***

| Output Standard | $V_{CCIO}$ (Typical) |
|---|---|
| **Single-Ended Interfaces** | |
| LVCMOS33 | 3.3 |
| LVCMOS25 | 2.5 |
| LVCMOS18 | 1.8 |

## On-Chip Strobe Generators

The iCE40LM devices feature two different Strobe Generators. One is tailored for low-power operation (Low Power Strobe Generator – LPSG), and generates periodic strobes in the Microsecond (µs) ranges. The other is tailored for high speed operation (High Speed Strobe Generator – HSSG), and generates periodic strobes in the Nanosecond (ns) ranges.Add a paragraph:

The Strobe Generators (HSSG and LPSG) provide fixed periodic strobes, and these strobes can be used as a clock source. When used as a clock source, the HSSG can provide strobe frequency in the range of 5MHz - 20MHz. The LPSG can provide strobe frequency in the range of 4KHz - 20KHz.

For further information on how to use the LPSG and HSSG, please refer to TN1275, iCE40LM On-Chip Strobe Generator Usage Guide.

## User I²C IP

The iCE40LM devices have two I²C IP cores. Either of the two cores can be configured either as an I²C master or as an I²C slave. Both I²C cores have preassigned pins, or user can select different pins, when the core is used.

When the IP core is configured as master, it will be able to control other devices on the I²C bus through the pre-assigned pin interface. When the core is configured as the slave, the device will be able to provide I/O expansion to an I²C Master. The I²C cores support the following functionality:

- Master and Slave operation
- 7-bit and 10-bit addressing
- Multi-master arbitration support
- Clock stretching
- Up to 400 KHz data transfer speed
- General Call support

For further information on the User I²C, please refer to TN1274, iCE40LM SPI/I2C Hardened IP Usage Guide.

## User SPI IP

The iCE40LM devices have two SPI IP cores. Both SPI cores have preassigned pins, or user can select different pins, when the SPI core is used. Both SPI IP core can be configured as a SPI master or as a slave. When the SPI IP core is configured as a master, it controls the other SPI enabled devices connected to the SPI Bus. When SPI IP core is configured as a slave, the device will be able to interface to an external SPI master.

The SPI IP core supports the following functions:

- Configurable Master and Slave modes
- Full-Duplex data transfer
- Mode fault error flag with CPU interrupt capability
- Double-buffered data register
- Serial clock with programmable polarity and phase
- LSB First or MSB First Data Transfer

For further information on the User SPI, please refer to TN1274, iCE40LM SPI/I2C Hardened IP Usage Guide.

## High Drive I/O Pins

The iCE40LM family devices offer 3 High Drive (HD) outputs in each device in the family. The HD outputs are ideal to drive LED signals on mobile application.

These HD outputs can be driven in different drive modes. The default is standard drive, which source/sink 8mA current nominally. When HD drive option is selected, these HD outputs can source/sink 24mA current nominally.

The pins on the HD I/Os are labeled with HD in it.

## Power On Reset

iCE40LM devices have power-on reset circuitry to monitor $V_{CC}$, $V_{CCIO\_2}$ and $V_{CC\_SPI}$ voltage levels during power-up and operation. At power-up, the POR circuitry monitors these voltage levels. It then triggers download from the external Flash memory after reaching the power-up levels specified in the Power-On-Reset Voltage table in the DC and Switching Characteristics section of this data sheet. Before and during configuration, the I/Os are held in tri-state. I/Os are released to user functionality once the device has finished configuration.

# iCE40LM Configuration

This section describes the programming and configuration of the iCE40LM family.

**Device Configuration**

There are various ways to configure the Configuration RAM (CRAM) including:

- From a SPI Flash (Master SPI mode)
- System microprocessor to drive a Serial Slave SPI port (SSPI mode)

For more details on configuring the iCE40LM, please see TN1248, iCE40 Programming and Configuration.

## Power Saving Options

The iCE40LM devices feature iCEGate and PLL low power mode to allow users to meet the static and dynamic power requirements of their applications. Table 9-9 describes the function of these features.

***Table 9-9. iCE40LM Power Saving Features Description***

| Device Subsystem | Feature Description |
|---|---|
| PLL | When LATCHINPUTVALUE is enabled, forces the PLL into low-power mode; PLL output held static at last input clock value. |
| iCEGate | To save power, the optional iCEGate latch can selectively freeze the state of individual, non-registered inputs within an I/O bank. Registered inputs are effectively frozen by their associated clock or clock-enable control. |

# iCE40LM Family Data Sheet
# DC and Switching Characteristics



## Absolute Maximum Ratings[1, 2, 3]

Supply Voltage $V_{CC}$ . . . . . -0.5 to 1.42V

Output Supply Voltage $V_{CCIO}$ and $V_{CC\_SPI}$ . . . . . -0.5 to 3.60V

PLL Power Supply, $V_{CCPLL}$ . . . . . -0.5 to 1.3V

I/O Tri-state Voltage Applied. . . . . -0.5 to 3.60V

Dedicated Input Voltage Applied . . . . . -0.5 to 3.60V

Storage Temperature (Ambient). . . . . -65°C to 150°C

Junction Temperature ($T_J$) . . . . . -55°C to 125°C

1. Stress above those listed under the "Absolute Maximum Ratings" may cause permanent damage to the device. Functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied.
2. Compliance with the Lattice Thermal Management document is required.
3. All voltages referenced to GND.

## Recommended Operating Conditions[1]

| Symbol | Parameter | | Min. | Max. | Units |
|---|---|---|---|---|---|
| $V_{CC}$[1] | Core Supply Voltage | | 1.14 | 1.26 | V |
| $V_{CCIO}$[1, 2, 3] | I/O Driver Supply Voltage | $V_{CCIO\_0}$, $V_{CCIO\_2}$ | 1.71 | 3.46 | V |
| $V_{CCPLL}$[4] | PLL Power Supply Voltage | | 1.14 | 1.26 | V |
| $V_{CC\_SPI}$[5] | Config SPI port Power Supply Voltage | | 1.71 | 3.46 | V |
| $t_{JIND}$ | Junction Temperature Industrial Operation | | -40 | 100 | °C |

1. Like power supplies must be tied together. $V_{CC}$ to $V_{CCPLL}$, $V_{CCIO\_0}$ to $V_{CCIO\_2}$ if they are at same supply voltage.
2. See recommended voltages by I/O standard in subsequent table.
3. $V_{CCIO}$ pins of unused I/O banks should be connected to the $V_{CC}$ power supply on boards.
4. For 25-pin WLCSP, PLL is not supported.
5. For 25-pin WLCSP, $V_{CC\_SPI}$ is connected to $V_{CCIO\_2}$ on the package. For all other packages, $V_{CC\_SPI}$ is used to power the SPI1 ports in both configuration mode and user mode.

## Power Supply Ramp Rates[1]

| Symbol | Parameter | Min. | Max. | Units |
|---|---|---|---|---|
| $t_{RAMP}$ | Power supply ramp rates for all power supplies. | 0.6 | 10 | V/ms |

1. Assumes monotonic ramp rates.

## Power-On-Reset Voltage Levels[1]

| Symbol | Parameter | | Min. | Max. | Units |
|---|---|---|---|---|---|
| $V_{PORUP}$ | Power-On-Reset ramp-up trip point (circuit monitoring $V_{CC}$, $V_{CCIO\_2}$ and $V_{CC\_SPI}$) | $V_{CC}$ | 0.64 | 0.99 | V |
| | | $V_{CCIO\_2}$, $V_{CC\_SPI}$ | 0.70 | 1.59 | V |
| $V_{PORDN}$ | Power-On-Reset ramp-down trip point (circuit monitoring $V_{CC}$, $V_{CCIO\_2}$ and $V_{CC\_SPI}$) | $V_{CC}$ | — | 0.66 | V |
| | | $V_{CCIO\_2}$, $V_{CC\_SPIl}$ | — | 1.59 | V |

1. These POR trip points are only provided for guidance. Device operation is only characterized for power supply voltages specified under recommended operating conditions.

## Power Up Sequence

For all iCE40LM devices, it is required to have the $V_{CC}$/$V_{CCPLL}$ power supply powered up before all other power supplies. The $V_{CC}$/$V_{CCPLL}$ has to be higher than 0.5V before other supplies are powered from GND.

Following $V_{CC}$/$V_{CCPLL}$, $V_{CCSPI}$ should be ramped up, followed by the remaining supplies. For 25-pin WLCSP, $V_{CC\_SPI}$ is connected to $V_{CCIO\_2}$, and the $V_{CCPLL}$ is internally connected for that package.

## ESD Performance

Please contact Lattice Semiconductor for additional information.

## DC Electrical Characteristics

**Over Recommended Operating Conditions**

| Symbol | Parameter | Condition | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $I_{IL}$, $I_{IH}$[1, 3, 4] | Input or I/O Leakage | $0V < V_{IN} < V_{CCIO} + 0.2V$ | — | — | +/-10 | µA |
| $C_1$ | I/O Capacitance[2] | $V_{CCIO}$ = 3.3V, 2.5V, 1.8V<br>$V_{CC}$ = Typ., $V_{IO}$ = 0 to $V_{CCIO}$ + 0.2V | — | 6 | — | pf |
| $C_2$ | Global Input Buffer Capacitance[2] | $V_{CCIO}$ = 3.3V, 2.5V, 1.8V<br>$V_{CC}$ = Typ., $V_{IO}$ = 0 to $V_{CCIO}$ + 0.2V | — | 6 | — | pf |
| $V_{HYST}$ | Input Hysteresis | $V_{CCIO}$ = 1.8V, 2.5V, 3.3V | — | 200 | — | mV |
| $I_{PU}$ | Internal PIO Pull-up Current | $V_{CCIO}$ = 1.8V, 0=<$V_{IN}$<=0.65 $V_{CCIO}$ | -3 | — | -31 | µA |
| | | $V_{CCIO}$ = 2.5V, 0=<$V_{IN}$<=0.65 $V_{CCIO}$ | -8 | — | -72 | µA |
| | | $V_{CCIO}$ = 3.3V, 0=<$V_{IN}$<=0.65 $V_{CCIO}$ | -11 | — | -128 | µA |

1. Input or I/O leakage current is measured with the pin configured as an input or as an I/O with the output driver tri-stated. It is not measured with the output driver active. Internal pull-up resistors are disabled.
2. $T_J$ 25°C, f = 1.0 MHz.
3. Please refer to $V_{IL}$ and $V_{IH}$ in the sysIO Single-Ended DC Electrical Characteristics table of this document.
4. Some products are clamped to a diode when $V_{IN}$ is larger than $V_{CCIO}$.

## Supply Current [1, 2, 3, 4]

| Symbol | Parameter | Typ. VCC[4] | Units |
|---|---|---|---|
| $I_{CCSTDBY}$ | Core Power Supply Static Current | 100 | µA |
| $I_{CCPLLSTDBY}$ | PLL Power Supply Static Current | | µA |
| $I_{CCIOSTDBY}$, $I_{CC\_SPISTDBY}$ | $V_{CCIO}$, $V_{CC\_SPI}$ Power Supply Static Current | | µA |
| $I_{CCPEAK}$ | Core Power Supply Startup Peak Current | | µA |
| $I_{CCPLLPEAK}$ | PLL Power Supply Startup Peak Current | | µA |
| $I_{CCIOPEAK}$, $I_{CC\_SPIPEAK}$ | $V_{CCIO}$, $V_{CC\_SPI}$ Power Supply Startup Peak Current | | µA |

1. Assumes blank pattern with the following characteristics: all outputs are tri-stated, all inputs are configured as LVCMOS and held at $V_{CCIO}$ or GND, on-chip PLL is off. For more detail with your specific design, use the Power Calculator tool. Power specified with master SPI configuration mode. Other modes may be up to 25% higher.
2. Frequency = 0 MHz.
3. TJ = 25°C, power supplies at nominal voltage.
4. Does not include pull-up.
5. For 25-pin WLCSP, $V_{CCPLL}$ is tied internally on the package, and $V_{CC\_SPI}$ is also connected to $V_{CCIO\_2}$ on the package.

## User I²C Specifications

| Parameter Symbol | Parameter Description | spec (STD Mode) | | | spec (FAST Mode) | | | Units |
|---|---|---|---|---|---|---|---|---|
| | | Min | Typ | Max | Min | Typ | Max | |
| $f_{SCL}$ | Maximum SCL clock frequency | — | — | 100 | — | — | 400 | KHz |
| $t_{HI}$ | SCL clock HIGH Time | 4 | — | — | 0.6 | — | — | us |
| $t_{LO}$ | SCL clock LOW Time | 4.7 | — | — | 1.3 | — | — | us |
| $t_{SU,DAT}$ | Setup time (DATA) | 250 | — | — | 100 | — | — | ns |
| $t_{HD,DAT}$ | Hold time (DATA) | 0 | — | — | 0 | — | — | ns |
| $t_{SU,STA}$ | Setup time (START condition) | 4.7 | — | — | 0.6 | — | — | us |
| $t_{HD,STA}$ | Hold time (START condition) | 4 | — | — | 0.6 | — | — | us |
| $t_{SU,STO}$ | Setup time (STOP condition) | 4 | — | — | 0.6 | — | — | us |
| $t_{BUF}$ | Bus free time between STOP and START | 4.7 | — | — | 1.3 | — | — | us |
| $t_{CO,DAT}$ | SCL LOW to DATAOUT valid | — | — | 3.4 | — | — | 0.9 | us |

## User SPI Specifications

| Parameter Symbol | Parameter Description | Min | Typ | Max | Units |
|---|---|---|---|---|---|
| $f_{MAX}$ | Maximum SCK clock frequency | — | — | 45 | MHz |
| $t_{HI}$ | HIGH period of SCK clock | 9 | — | — | ns |
| $t_{LO}$ | LOW period of SCK clock | 9 | — | — | ns |
| $t_{SUmaster}$ | Setup time (master mode) | 2 | — | — | ns |
| $t_{HOLDmaster}$ | Hold time (master mode) | 5 | — | — | ns |
| $t_{SUslave}$ | Setup time (slave mode) | 2 | — | — | ns |
| $t_{HOLDslave}$ | Hold time (slave mode) | 5 | — | — | ns |
| $t_{SCK2OUT}$ | SCK to out (slave mode) | — | — | 13.5 | ns |

## sysIO Recommended Operating Conditions

| Standard | $V_{CCIO}$ (V) | | |
|---|---|---|---|
| | Min. | Typ. | Max. |
| LVCMOS 3.3 | 3.14 | 3.3 | 3.46 |
| LVCMOS 2.5 | 2.37 | 2.5 | 2.62 |
| LVCMOS 1.8 | 1.71 | 1.8 | 1.89 |

## sysIO Single-Ended DC Electrical Characteristics

| Input/ Output Standard | $V_{IL}$ | | $V_{IH}$[1] | | $V_{OL}$ Max. (V) | $V_{OH}$ Min. (V) | $I_{OL}$ Max. (mA) | $I_{OH}$ Max. (mA) |
|---|---|---|---|---|---|---|---|---|
| | Min. (V) | Max. (V) | Min. (V) | Max. (V) | | | | |
| LVCMOS 3.3 | -0.3 | 0.8 | 2.0 | $V_{CCIO}$ + 0.2V | 0.4 | $V_{CCIO}$ - 0.5 | 8 | -8 |
| | | | | | 0.2 | $V_{CCIO}$ - 0.2 | 0.1 | -0.1 |
| LVCMOS 2.5 | -0.3 | 0.7 | 1.7 | $V_{CCIO}$ + 0.2V | 0.4 | $V_{CCIO}$ - 0.5 | 6 | -6 |
| | | | | | 0.2 | $V_{CCIO}$ - 0.2 | 0.1 | -0.1 |
| LVCMOS 1.8 | -0.3 | $0.35V_{CCIO}$ | $0.65V_{CCIO}$ | $V_{CCIO}$ + 0.2V | 0.4 | $V_{CCIO}$ - 0.4 | 4 | -4 |
| | | | | | 0.2 | $V_{CCIO}$ - 0.2 | 0.1 | -0.1 |

1. Some products are clamped to a diode when $V_{IN}$ is larger than $V_{CCIO}$.

## Typical Building Block Function Performance[1, 2]

### Pin-to-Pin Performance (LVCMOS25)

| Function | Timing | Units |
|---|---|---|
| **Basic Functions** | | |
| 16-bit decoder | 16.5 | ns |
| 4:1 MUX | 18.0 | ns |
| 16:1 MUX | 19.5 | ns |

### Register-to-Register Performance

| Function | Timing | Units |
|---|---|---|
| **Basic Functions** | | |
| 16:1 MUX | 110 | MHz |
| 16-bit adder | 100 | MHz |
| 16-bit counter | 100 | MHz |
| 64-bit counter | 40 | MHz |
| **Embedded Memory Functions** | | |
| 256x16 Pseudo-Dual Port RAM | 150 | MHz |

1. The above timing numbers are generated using the iCECube2 design tool. Exact performance may vary with device and tool version. The tool uses internal parameters that have been characterized but are not tested on every device.
2. Using a $V_{CC}$ of 1.14V at Junction Temp 85C.

## Derating Logic Timing

Logic timing provided in the following sections of the data sheet and the Lattice design tools are worst case numbers in the operating range. Actual delays may be much faster. Lattice design tools can provide logic timing numbers at a particular temperature and voltage.

## Maximum sysIO Buffer Performance[1]

| I/O Standard | Max. Speed | Units |
|---|---|---|
| **Inputs** | | |
| LVCMOS33 | | MHz |
| LVCMOS25 | | MHz |
| LVCMOS18 | | MHz |
| **Outputs** | | |
| LVCMOS33 | | MHz |
| LVCMOS25 | | MHz |
| LVCMOS18 | | MHz |

1. Measured with a toggling pattern

## iCE40LM Family Timing Adders

**Over Recommended Commercial Operating Conditions[1, 2, 3]**

| Buffer Type | Description | Timing | Units |
|---|---|---|---|
| **Input Adjusters** | | | |
| LVCMOS33 | LVCMOS, $V_{CCIO}$ = 3.3V | 0.18 | nS |
| LVCMOS25 | LVCMOS, $V_{CCIO}$ = 2.5V | 0.00 | nS |
| LVCMOS18 | LVCMOS, $V_{CCIO}$ = 1.8V | 0.19 | nS |
| **Output Adjusters** | | | |
| LVCMOS33 | LVCMOS, $V_{CCIO}$ = 3.3V | -0.12 | nS |
| LVCMOS25 | LVCMOS, $V_{CCIO}$ = 2.5V | 0.00 | nS |
| LVCMOS18 | LVCMOS, $V_{CCIO}$ = 1.8V | 1.32 | nS |

1. Timing adders are relative to LVCMOS25 and characterized but not tested on every device.
2. LVCMOS timing measured with the load specified in Switching Test Condition table.
3. Commercial timing numbers are shown.

# iCE40LM External Switching Characteristics

**Over Recommended Commercial Operating Conditions**

| Parameter | Description | Device | | | Units |
|---|---|---|---|---|---|
| **Clocks** | | | | | |
| **Global Clocks** | | | | | |
| $f_{MAX\_GBUF}$ | Frequency for Global Buffer Clock network | All devices | | 185 | MHz |
| $t_{W\_GBUF}$ | Clock Pulse Width for Global Buffer | All devices | TBD | — | ns |
| $t_{SKEW\_GBUF}$ | Global Buffer Clock Skew Within a Device | All devices | — | 650 | ps |
| **Pin-LUT-Pin Propagation Delay** | | | | | |
| $t_{PD}$ | Best case propagation delay through one LUT logic | All devices | — | 14.0 | ns |
| **General I/O Pin Parameters (Using Global Buffer Clock without PLL)** | | | | | |
| $t_{SKEW\_IO}$ | Data bus skew across a bank of IOs | All devices | — | 400 | ps |
| $t_{CO}$ | Clock to Output - PIO Output Register | All devices | — | 9.0 | ns |
| $t_{SU}$ | Clock to Data Setup - PIO Input Register | All devices | | — | ns |
| $t_{H}$ | Clock to Data Hold - PIO Input Register | All devices | 5.55 | — | ns |
| **General I/O Pin Parameters (Using Global Buffer Clock with PLL)[1]** | | | | | |
| $t_{CO}$ | Clock to Output - PIO Output Register | All devices | — | TBD | ns |
| $t_{SU}$ | Clock to Data Setup - PIO Output Register | All devices | TBD | — | ns |
| $t_{H}$ | Clock to Data Hold - PIO Output Register | All devices | TBD | — | ns |

1. 25-pin WLCSP package does not support PLL.

# sysCLOCK PLL Timing – Preliminary

**Over Recommended Operating Conditions**

| Parameter | Descriptions | Conditions | Min. | Max. | Units |
|---|---|---|---|---|---|
| $f_{IN}$ | Input Clock Frequency (REFERENCECLK, EXTFEEDBACK) | | TBD | TBD | MHz |
| $f_{OUT}$ | Output Clock Frequency (PLLOUT) | | TBD | TBD | MHz |
| $f_{VCO}$ | PLL VCO Frequency | | TBD | TBD | MHz |
| $f_{PFD}$ | Phase Detector Input Frequency | | TBD | TBD | MHz |
| **AC Characteristics** | | | | | |
| $t_{DT}$ | Output Clock Duty Cycle | | | TBD | % |
| $t_{PH}$ | Output Phase Accuracy | | — | TBD | deg |
| $t_{OPJIT}$[1, 5] | Output Clock Period Jitter | $f_{OUT}$ <= 100 MHz | — | TBD | ps p-p |
| | | $f_{OUT}$ > 100 MHz | — | TBD | UIPP |
| | Output Clock Cycle-to-cycle Jitter | $f_{OUT}$ <= 100 MHz | — | TBD | ps p-p |
| | | $f_{OUT}$ > 100 MHz | — | TBD | UIPP |
| | Output Clock Phase Jitter | $f_{PFD}$ <= 25 MHz | — | TBD | ps p-p |
| | | $f_{PFD}$ > 25 MHz | — | TBD | UIPP |
| $t_W$ | Output Clock Pulse Width | At 90% or 10% | | TBD | ns |
| $t_{LOCK}$[2, 3] | PLL Lock-in Time | | — | TBD | us |
| $t_{UNLOCK}$ | PLL Unlock Time | | — | TBD | ns |
| $t_{IPJIT}$[4] | Input Clock Period Jitter | $f_{PFD} \geq 20$ MHz | — | TBD | ps p-p |
| | | $f_{PFD}$ < 20 MHz | — | TBD | UIPP |
| $t_{FDTAP}$ | Fine Delay adjustment, per Tap | | — | TBD | ps |
| $t_{STABLE}$[3] | LATCHINPUTVALUE LOW to PLL Stable | | — | TBD | ns |
| $t_{STABLE\_PW}$[3] | LATCHINPUTVALUE Pulse Width | | — | TBD | ns |
| $t_{RST}$ | RESET Pulse Width | | TBD | TBD | ns |
| $t_{RSTREC}$ | RESET Recovery Time | | TBD | TBD | ns |
| $t_{DYNAMIC\_WD}$ | DYNAMICDELAY Pulse Width | | TBD | TBD | VCO Cycles |

1. Period jitter sample is taken over 10,000 samples of the primary PLL output with a clean reference clock. Cycle-to-cycle jitter is taken over 1000 cycles. Phase jitter is taken over 2000 cycles. All values per JESD65B.
2. Output clock is valid after $t_{LOCK}$ for PLL reset and dynamic delay adjustment.
3. At minimum $f_{PFD}$. As the $f_{PFD}$ increases the time will decrease to approximately 60% the value listed.
4. Maximum limit to prevent PLL unlock from occurring. Does not imply the PLL will operate within the output specifications listed in this table.
5. The jitter values will increase with loading of the PLD fabric and in the presence of SSO noise.

## SPI Master Configuration Time[1]

| Symbol | Parameter | Conditions | Max. | Units |
|---|---|---|---|---|
| $t_{CONFIG}$ | POR/CRESET_B to Device I/O Active | All devices - Low Frequency (Default) | 70 | ms |
| | | All devices - Medium frequency | 35 | ms |
| | | All devices - High frequency | 18 | ms |

1. Assumes sysMEM Block is initialized to an all zero pattern if they are used.

## sysCONFIG Port Timing Specifications

| Symbol | Parameter | Conditions | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| **All Configuration Modes** | | | | | | |
| $t_{CRESET\_B}$ | Minimum CRESET_B LOW pulse width required to restart configuration, from falling edge to rising edge | | 200 | — | — | ns |
| $t_{DONE\_IO}$ | Number of configuration clock cycles after CDONE goes HIGH before the PIO pins are activated | | 49 | — | — | Cycles |
| **Slave SPI** | | | | | | |
| $t_{CR\_SCK}$ | Minimum time from a rising edge on CRESET_B until the first SPI WRITE operation, first SPI_XCK clock. During this time, the iCE40LM device is clearing its internal configuration memory | | 1200 | — | — | us |
| $f_{MAX}$ | CCLK clock frequency | Write | 1 | — | 25 | MHz |
| | | Read[1] | — | 15 | — | MHz |
| $t_{CCLKH}$ | CCLK clock pulsewidth HIGH | | 20 | — | — | ns |
| $t_{CCLKL}$ | CCLK clock pulsewidth LOW | | 20 | — | — | ns |
| $t_{STSU}$ | CCLK setup time | | 12 | — | — | ns |
| $t_{STH}$ | CCLK hold time | | 12 | — | — | ns |
| $t_{STCO}$ | CCLK falling edge to valid output | | 13 | — | — | ns |
| **Master SPI** | | | | | | |
| $f_{MCLK}$ | MCLK clock frequency | Low Frequency (Default) | 6 | | | MHz |
| | | Medium Frequency[2] | 18 | | | MHz |
| | | High Frequency[2] | 31 | | | MHz |
| $t_{MCLK}$ | CRESET_B HIGH to first MCLK edge | | 1200 | — | — | us |

1. Supported with 1.2V Vcc and at 25C.
2. Extended range $f_{MAX}$ Write operation support up to 53MHz with 1.2V $V_{CC}$ and at 25C.

## Switching Test Conditions

Figure 10-1 shows the output test load used for AC testing. The specific values for resistance, capacitance, voltage, and other test conditions are shown in Table 10-1.

***Figure 10-1. Output Test Load, LVCMOS Standards***

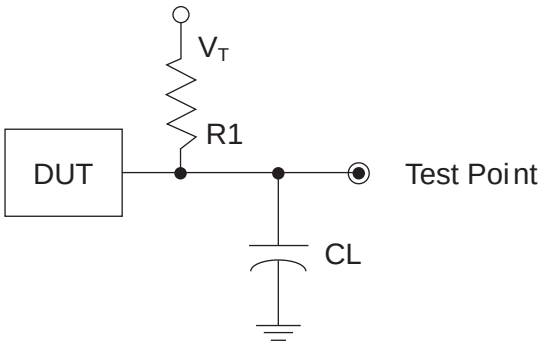



***Table 10-1. Test Fixture Required Components, Non-Terminated Interfaces***

| Test Condition | $R_1$ | $C_L$ | Timing Reference | $V_T$ |
|---|---|---|---|---|
| LVCMOS settings (L -> H, H -> L) | ∞ | 0 pF | LVCMOS 3.3 = 1.5V | — |
| | | | LVCMOS 2.5 = $V_{CCIO}/2$ | — |
| | | | LVCMOS 1.8 = $V_{CCIO}/2$ | — |
| LVCMOS 3.3 (Z -> H) | 188 | 0 pF | 1.5 | $V_{OL}$ |
| LVCMOS 3.3 (Z -> L) | | | 1.5 | $V_{OH}$ |
| Other LVCMOS (Z -> H) | | | $V_{CCIO}/2$ | $V_{OL}$ |
| Other LVCMOS (Z -> L) | | | $V_{CCIO}/2$ | $V_{OH}$ |
| LVCMOS (H -> Z) | | | $V_{OH}$ - 0.15 | $V_{OL}$ |
| LVCMOS (L -> Z) | | | $V_{OL}$ - 0.15 | $V_{OH}$ |

Note: Output test conditions for all other interfaces are determined by the respective standards.

## Signal Descriptions

| Signal Name | Function | I/O | Description |
|---|---|---|---|
| **Power Supplies** | | | |
| $V_{CC}$ | Power | - | Core Power Supply |
| $V_{CCIO\_0}$ | Power | - | Power Supply for I/Os in Bank 0 |
| $V_{CCIO\_2}$ | Power | - | Power Supply for I/Os in Bank 2 |
| $V_{CC\_SPI}$ | Power | - | Power supply for SPI1 ports. For 25-pin WLCSP, this signal is connected to $V_{CCIO\_2}$ |
| $V_{CCPLL}$ | Power | - | Power supply for PLL. For 25-pin WLCSP, this is connected internally to $V_{CC}$ |
| GND/GNDPLL | GROUND | - | Ground |
| **Dedicated Configuration Signals** | | | |
| CRESET | Configuration | I | Configuration Reset, active LOW. No internal pull-up resistor. Either actively driven externally or connect an 10K-ohm pull-up resistor to $V_{CCIO\_2}$. |
| CDONE | Configuration | I/O | Configuration Done. Includes a weak pull-up resistor to $V_{CCIO\_2}$. |
| **SPI and Config SPI Ports** | | | |
| SPI1_SCK/PIO[T/B]_x[HD] | User SPI1 | I/O | In user mode, used as CLK signal on SPI interface for sensor management function. |
| | Configuration | I/O | This pins is shared with device configuration. During configuration, this pin is CLK signal connecting to external SPI memory |
| | General I/O | I/O | In user mode, when the SPI interface is not used, user can program this pin as general I/O pin for user functions. (x represents ball on the package)<br>[T/B]: T=Vccio_0 bank, B=Vccio_2 bank.<br>[HD]=High Drive I/O |
| SPI1_MISO/PIO[T/B]_x[HD] | User SPI1 | I/O | In user mode, used as Input (Master Mode) or Output (Slave Mode) signal on SPI interface for sensor management function. |
| | Configuration | I/O | This pins is shared with device configuration. During configuration, this pin is Input signal connecting to external SPI memory. |
| | General I/O | I/O | In user mode, when the SPI interface is not used, user can program this pin as general I/O pin for user functions. (x represents ball on the package)<br>[T/B]: T=Vccio_0 bank, B=Vccio_2 bank.<br>[HD]=High Drive I/O |
| SPI1_MOSI/PIO[T/B]_x[HD] | User SPI1 | I/O | In user mode, used as Output (Master Mode) or Input (Slave Mode) signal on SPI interface for sensor management function. |
| | Configuration | I/O | This pins is shared with device configuration. During configuration, this pin is Output signal connecting to external SPI memory. |
| | General I/O | I/O | In user mode, when the SPI interface is not used, user can program this pin as general I/O pin for user functions. (x represents ball on the package)<br>[T/B]: T=Vccio_0 bank, B=Vccio_2 bank.<br>[HD]=High Drive I/O |

| Signal Name | Function | I/O | Description |
|---|---|---|---|
| SPI1_CSN/PIO[T/B]_x[HD] | User SPI1 | I/O | In user mode, used as CSN signal on SPI interface for sensor management function. This pin is output pin in Master Mode, and input in in Slave Mode. |
| | Configuration | I/O | This pins is shared with device configuration. During configuration, this pin is CSN signal connecting to external SPI memory |
| | General I/O | I/O | In user mode, when the SPI interface is not used, user can program this pin as general I/O pin for user functions. (x represents ball on the package)<br>[T/B]: T=Vccio_0 bank, B=Vccio_2 bank.<br>[HD]=High Drive I/O |
| SPI2_SCK/PIO[T/B]_x[HD] | User SPI2 | I/O | In user mode, used as CLK signal on SPI interface for sensor management function. |
| | General I/O | I/O | In user mode, when the SPI interface is not used, user can program this pin as general I/O pin for user functions. (x represents ball on the package)<br>[T/B]: T=Vccio_0 bank, B=Vccio_2 bank.<br>[HD]=High Drive I/O |
| SPI2_MISO/PIO[T/B]_x[HD] | User SPI2 | I/O | In user mode, used as Input (Master Mode) or Output (Slave Mode) signal on SPI interface for sensor management function. |
| | General I/O | I/O | In user mode, when the SPI interface is not used, user can program this pin as general I/O pin for user functions. (x represents ball on the package)<br>[T/B]: T=Vccio_0 bank, B=Vccio_2 bank.<br>[HD]=High Drive I/O |
| SPI2_MOSI/PIO[T/B]_x[HD] | User SPI2 | I/O | In user mode, used as Output (Master Mode) or Input (Slave Mode) signal on SPI interface for sensor management function. |
| | General I/O | I/O | In user mode, when the SPI interface is not used, user can program this pin as general I/O pin for user functions. (x represents ball on the package)<br>[T/B]: T=Vccio_0 bank, B=Vccio_2 bank.<br>[HD]=High Drive I/O |
| SPI2_CSN/PIO[T/B]_x[HD] | User SPI2 | I/O | In user mode, used as CSN signal on SPI interface for sensor management function. This pin is output pin in Master Mode, and input in in Slave Mode. |
| | General I/O | I/O | In user mode, when the SPI interface is not used, user can program this pin as general I/O pin for user functions. (x represents ball on the package)<br>[T/B]: T=Vccio_0 bank, B=Vccio_2 bank.<br>[HD]=High Drive I/O |
| **$I^2C$ Ports** | | | |
| I2C1_SCL/PIO[T/B]_x[HD] | User I2C1 | I/O | Used as CLK signal on $I^2C$ interface for sensor management function. |
| | General I/O | I/O | When the $I^2C$ interface is not used, user can program this pin as general I/O pin for user functions. (x represents ball on the package)<br>[T/B]: T=Vccio_0 bank, B=Vccio_2 bank.<br>[HD]=High Drive I/O |
| I2C1_SDA/PIO[T/B]_x[HD] | User I2C1 | I/O | Used as Data signal on $I^2C$ interface for sensor management function. |
| | General I/O | I/O | When the $I^2C$ interface is not used, user can program this pin as general I/O pin for user functions. (x represents ball on the package)<br>[T/B]: T=Vccio_0 bank, B=Vccio_2 bank.<br>[HD]=High Drive I/O |

| Signal Name | Function | I/O | Description |
|---|---|---|---|
| I2C2_SCL/PIO[T/B]_x[HD] | User I2C2 | I/O | Used as CLK signal on I²C interface for sensor management function. |
| | General I/O | I/O | When the I²C interface is not used, user can program this pin as general I/O pin for user functions. (x represents ball on the package)<br>[T/B]: T=Vccio_0 bank, B=Vccio_2 bank.<br>[HD]=High Drive I/O |
| I2C2_SDA/PIO[T/B]_x[HD] | User I2C2 | I/O | Used as Data signal on I²C interface for sensor management function. |
| | General I/O | I/O | When the I²C interface is not used, user can program this pin as general I/O pin for user functions. (x represents ball on the package)<br>[T/B]: T=Vccio_0 bank, B=Vccio_2 bank.<br>[HD]=High Drive I/O |
| **Global Signals** | | | |
| PIO[T/B]_x[HD]/Gn | General I/O | I/O | User can program this pin as general I/O pin for user functions. (x represents ball on the package)<br>[T/B]: T=Vccio_0 bank, B=Vccio_2 bank.<br>[HD]=High Drive I/O |
| | Global Signal | I | Global input used for high fanout, or clock/reset net. n=0,1,2,3,6,7. The Gn Global input pin can drive the corresponding GBUFn global buffer. |
| **General Purpose I/O** | | | |
| PIO[T/B]_x[HD] | General I/O | I/O | User can program this pin as general I/O pin for user functions. (x represents ball on the package) |

## Pin Information Summary

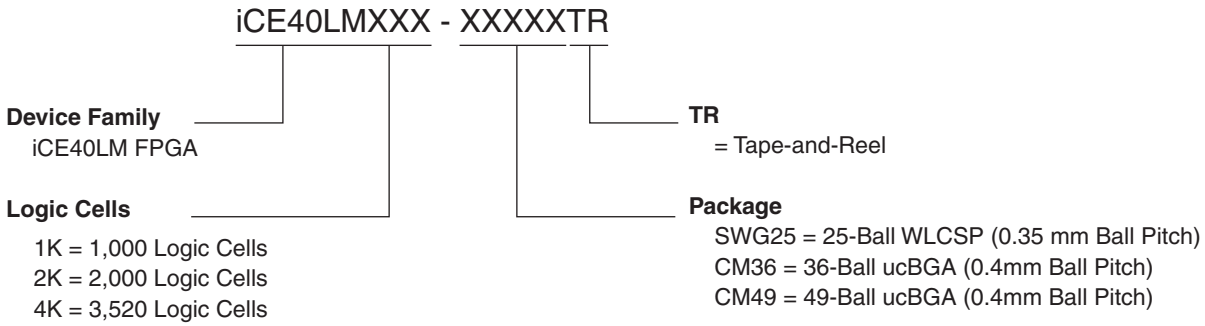
| Pin Type | | iCE40LM-1K | | | iCE40LM-2K | | | iCE40LM-4K | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | SWG25 | CM36 | CM49 | SWG25 | CM36 | CM49 | SWG25 | CM36 | CM49 |
| General Purpose I/O Per Bank | Bank 0 | 7 | 15 | 20 | 7 | 15 | 20 | 7 | 15 | 20 |
| | Bank 2[1] | 11 | 13 | 17 | 11 | 13 | 17 | 11 | 13 | 17 |
| Total General Purpose I/Os | | 18 | 28 | 37 | 18 | 28 | 37 | 18 | 28 | 37 |
| Vcc | | 1 | 1 | 2 | 1 | 1 | 2 | 1 | 1 | 2 |
| Vccio | Bank 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | Bank 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| $V_{CC\_SPI}$ | | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| $V_{CCPLL}$ | | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| Miscellaneous Dedicated Pins | | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| GND | | 2 | 2 | 4 | 2 | 2 | 4 | 2 | 2 | 4 |
| NC | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Reserved | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Total Balls | | 25 | 36 | 49 | 25 | 36 | 49 | 25 | 36 | 49 |
| SPI Interfaces | Bank 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Bank 2 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 |
| I²C Interfaces | Bank 0 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 |
| | Bank 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

1. Including General Purpose I/Os powered by $V_{CC\_SPI}$ and $V_{CCPLL}$.

# iCE40LM Family Data Sheet Ordering Information



## iCE40LM Part Number Description




All parts are shipped in tape-and-reel.

## Ordering Part Numbers

| Part Number | LUTs | Supply Voltage | Package | Leads | Temp. |
|---|---|---|---|---|---|
| iCE40LM1K-SWG25TR | 1000 | 1.2V | Halogen-Free caBGA | 25 | IND |
| iCE40LM1K-CM36TR | 1000 | 1.2V | Halogen-Free csBGA | 36 | IND |
| iCE40LM1K-CM49TR | 1000 | 1.2V | Halogen-Free csBGA | 49 | IND |
| iCE40LM2K-SWG25TR | 2000 | 1.2V | Halogen-Free caBGA | 25 | IND |
| iCE40LM2K-CM36TR | 2000 | 1.2V | Halogen-Free csBGA | 36 | IND |
| iCE40LM2K-CM49TR | 2000 | 1.2V | Halogen-Free csBGA | 49 | IND |
| iCE40LM4K-SWG25TR | 3520 | 1.2V | Halogen-Free caBGA | 25 | IND |
| iCE40LM4K-CM36TR | 3520 | 1.2V | Halogen-Free csBGA | 36 | IND |
| iCE40LM4K-CM49TR | 3520 | 1.2V | Halogen-Free csBGA | 49 | IND |

# iCE40LM Family Data Sheet Supplemental Information



## For Further Information

A variety of technical notes for the iCE40 family are available on the Lattice web site.

- TN1248, iCE40 Programming and Configuration
- TN1274, iCE40LM SPI/I2C Hardened IP Usage Guide
- TN1275, iCE40LM On-Chip Strobe Generator Usage Guide
- TN1276, iCE40LM Advanced SPI/I2C Hardened IP Usage Guide
- TN1250, Memory Usage Guide for iCE40 Devices
- TN1251, iCE40 sysCLOCK PLL Design and Usage Guide
- iCE40LM Pinout Files
- iCE40LM Pin Migration Files
- Thermal Management document
- Lattice design tools
- Schematic Symbols

# iCE40LM Family Data Sheet Revision History

**October 2013** **Advance Data Sheet DS1045**

| Date | Version | Section | Change Summary |
|---|---|---|---|
| August 2013 | 00.1 EAP | All | Initial release. |
| September 2013 | 00.2 EAP | All | General updates to all sections. |
| October 2013 | 01.0 | All | General updates for product launch. |
| | | Pinout Information | Updated ball map to 25-pin WLCSP. |

# Section III. iCE40 LP/HX/LM Family Technical Notes

# iCE40 Programming and Configuration

October 2013 Technical Note TN1248

## Introduction

The iCE40™ devices are SRAM-based FPGAs. iCE40 LP and iCE40 HX devices have an on-chip, one-time programmable NVCM (Non-volatile Configuration Memory) to store configuration data. The SRAM memory cells are volatile, meaning that once power is removed from the device, its configuration is lost, and must be reloaded on the next power-up. This behavior has the advantage of being re-programmable in the field which provides flexibility for products already deployed to the field. But it also requires that the configuration information be stored in a non-volatile device and loaded each time power is applied to the device. The on-chip NVCM allows the device to configure instantly and greatly enhances the design security by eliminating the need to use an external memory device. The configuration data can also be stored in an external SPI Flash from which the FPGA can configure itself upon power-up. This is useful for prototyping the FPGA or in situations where reconfigurability is required. Additionally, the device can be configured by a processor in an embedded environment.

***Table 15-1. iCE40 Devices Configuration Features Comparison***

| Features | iCE40LM | iCE40 LP/iCE40 HX |
|---|---|---|
| NVCM (one time programmable) | - | X |
| Multiple Configuration Image | - | X |
| Master SPI Configuration Mode | X | X |
| Slave SPI Configuration Mode | X | X |

## Configuration Overview

The iCE40 LP and iCE40 HX devices contain two types of memory, SRAM and NVCM (one-time programmable). The iCE40LM devices, however, contain only one type of memory, which is SRAM. The SRAM memory contains the active configuration. The NVCM and the external SPI Flash provides a non-volatile storage for the configuration data. Additionally, the iCE40 configuration data can be downloaded from an external processor, microcontroller, or DSP processor using the SPI interface. In this document, the term “programming” refers to the programming of the NVCM and the term “configuration” refers to the configuration of SRAM memory. For either programming or configuration, the iCE40 FPGA utilizes the SPI configuration interface.

As described in Table 15-2, iCE40 components are configured for a specific application by loading a binary configuration bitstream image, generated by the Lattice development system. For high-volume applications, the bitstream image is usually permanently programmed in the on-chip Nonvolatile Configuration Memory (NVCM). However, the bitstream image can also be stored externally in a standard, low-cost commodity SPI serial Flash PROM. The iCE40 component can automatically load the image using the SPI Master Configuration Interface. Similarly, the iCE40 configuration data can be downloaded from an external processor, microcontroller, or DSP processor using a SPI-like serial interface.

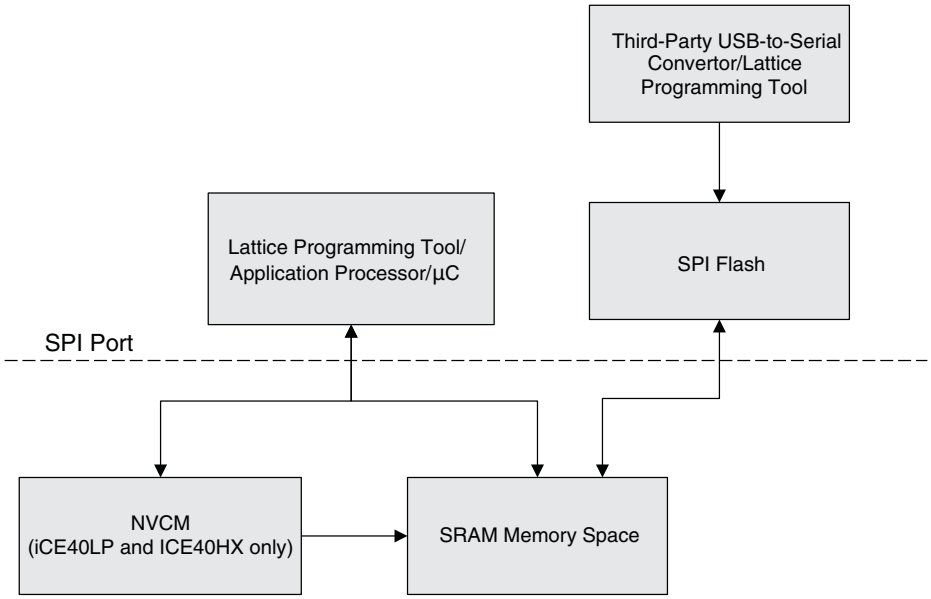
***Table 15-2. iCE40 Device Configuration Modes***

| Mode | Analogy | Configuration Data Source |
|---|---|---|
| NVCM[1] | ASIC | Internal, lowest-cost, secure, one-time programmable Nonvolatile Configuration Memory (NVCM). |
| Master SPI | Microprocessor | External, low-cost, commodity, SPI serial Flash PROM. |
| Slave SPI | Processor Peripheral | Configured by external device, such as a processor, microcontroller, or DSP using practically any data source, such as system Flash, a disk image, or over a network connection. |

1. iCE40 LP and iCE40 HX devices only.

Figure 15-1 provides an overview of the configuration and programming of the iCE40 FPGA. For configuration and programming, the device can be accessed using the SPI interface/protocol described in later sections of this technical note. The SRAM can configure itself (device in master mode) from the on-chip NVCM (iCE40 LP and ICE40 HX devices only), external SPI Flash or Lattice programming hardware. An external processor or programming hardware can also configure the SRAM with the FPGA in slave SPI mode. The NVCM can be programmed using the Lattice Diamond® Programmer (version 2.2 or later) or an external processor.

***Figure 15-1. Configuring and Programming the iCE40 Device***



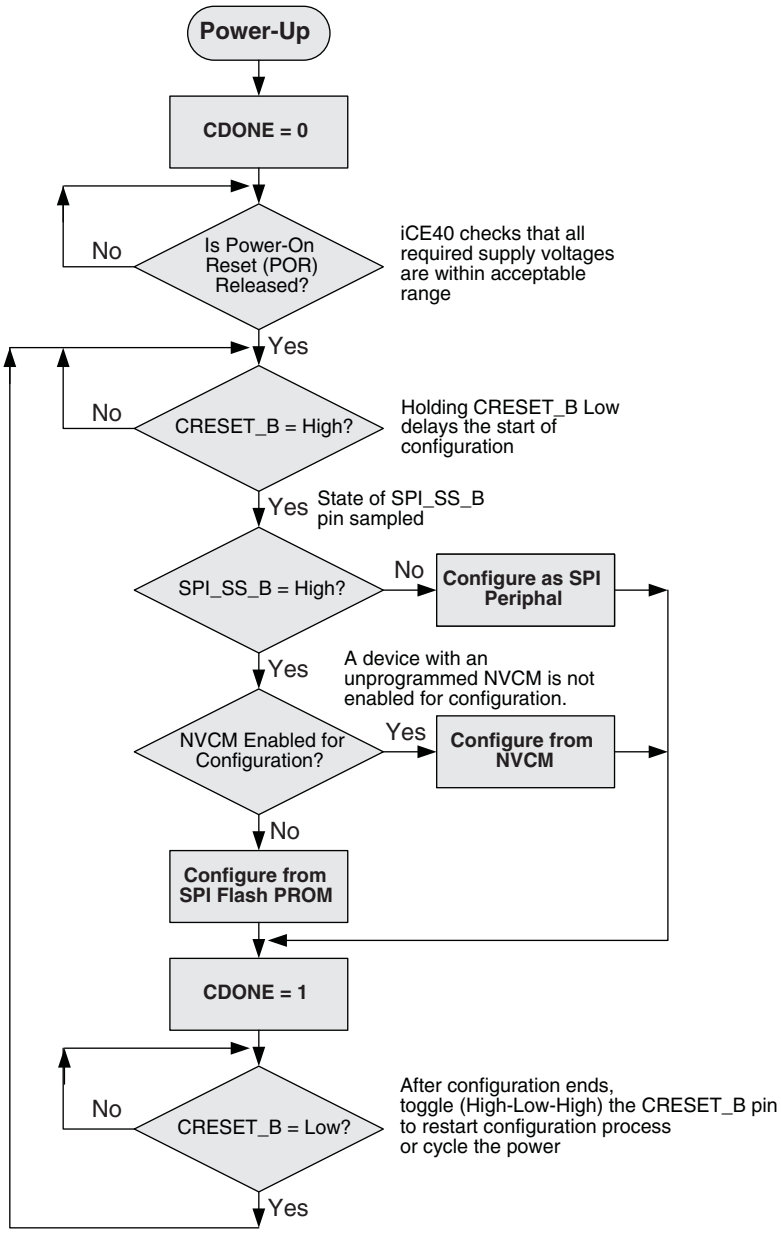

# Configuration Mode Selection

## Mode Selection for iCE40 LP and iCE40 HX

The iCE40 LP or iCE40 HX configuration mode is selected according to the following priority described below and illustrated in Figure 15-2.

- After exiting the Power-On Reset (POR) state or when CRESET_B returns High after being held Low, the iCE40 device samples the logical value on its SPI_SS_B pin. Like other programmable I/O pins, the SPI_SS_B pin has an internal pull-up resistor. Refer to the iCE40 LP/HX Family Data Sheet for the minimum pulse width requirement of CRESET_B.
- If the SPI_SS_B pin is sampled as a logic '1' (High), then …
  - Check if the device is enabled to configure from the Nonvolatile Configuration Memory (NVCM). If the NVCM is programmed, the device will configure from NVCM.
    - If enabled to configure from NVCM, the device configures itself using the Nonvolatile Configuration Memory (NVCM).
    - If not enabled to configure from NVCM, then the device configures using the SPI Master Configuration Interface.
- If the SPI_SS_B pin is sampled as a logic '0' (Low), then the device waits to be configured from an external controller or from another device in SPI Master Configuration Mode using an SPI-like interface.

*Figure 15-2. iCE40 LP and iCE40 HX Device Configuration Control Flow*

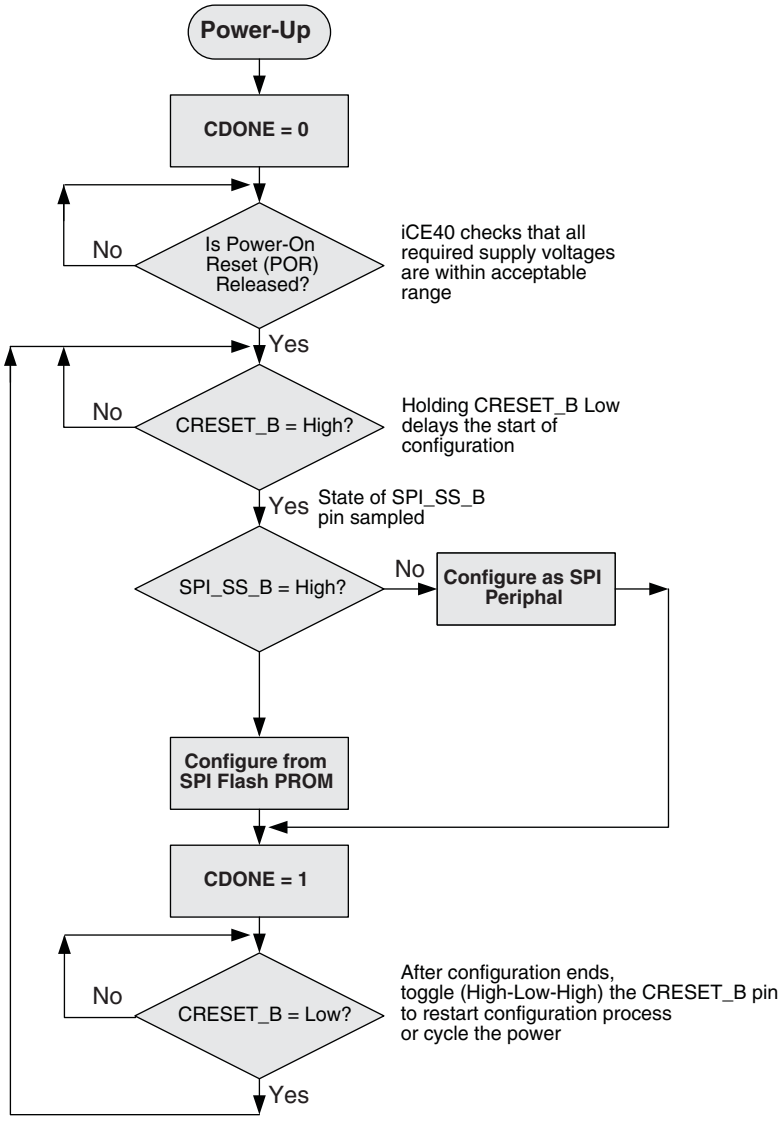


## Mode Selection for iCE40LM

The iCE40LM configuration mode is selected according to the following priority described below and illustrated in Figure 15-3.

- After exiting the Power-On Reset (POR) state or when CRESET_B returns High after being held Low, the device samples the logical value on its SPI_SS_B pin. Like other programmable I/O pins, the SPI_SS_B pin has an internal pull-up resistor. Refer to the iCE40LM Family Data Sheet for the minimum pulse width requirement of CRESET_B.
- If the SPI_SS_B pin is sampled as a logic '1' (High), then the device configures using the SPI Master Configuration Interface.
- If the SPI_SS_B pin is sampled as a logic '0' (Low), then the device waits to be configured from an external controller or from another device in SPI Master Configuration Mode using an SPI-like interface

***Figure 15-3. iCE40LM Device Configuration Control Flow***



## Nonvolatile Configuration Memory (NVCM) *(Applies to iCE40 LP and iCE40 HX Devices Only)*

All standard iCE40 devices have an internal, nonvolatile configuration memory (NVCM). The NVCM is large enough to program a complete iCE40 device, including initializing all Embedded Block RAM. The NVCM memory also has very high programming yield due to extensive error checking and correction (ECC) circuitry.

The NVCM is ideal for cost-sensitive, high-volume production applications, saving the cost and board space associated with an external configuration PROM. Furthermore, the NVCM provides exceptional design security, protecting critical intellectual property (IP). The NVCM contents are entirely contained within the iCE40 device and are not readable once protected by the one-time programmable Security bits. Furthermore, there is no observable difference between a programmed or un-programmed memory cell using optical or electron microscopy.

The NVCM memory has a programming interface similar to a 25-series SPI serial Flash PROM. Consequently, it can be programmed using Diamond Programmer (version 2.2 or later) before or after circuit board assembly or programmed in-system from a microprocessor or other intelligent controller. The NVCM can also be pre-programmed at the factory. Contact Lattice Technical Support or your local Lattice sales office for assistance.

### NVCM Programming

The NVCM can be programmed in the following ways:

- Diamond Programmer
  - Programming using the Diamond Programmer (Diamond 2.0.1 or later) is recommended for prototyping. Programming is supported using the Lattice programming cable. For more information refer to the Diamond Programmer Online Help and UG48, Lattice ispDOWNLOAD Cable User's Guide.
- Factory Programming
  - The Lattice factory offers NVCM programming. For more information contact your local Lattice sales office.
- Embedded Programming

The NVCM can be programmed using a processor. For more information contact your local Lattice sales office.

## Configuration Control Signals

The iCE40 configuration process is self-timed and controlled by a few internal signals and device I/O pins, as described in Table 15-3.

***Table 15-3. iCE40 Configuration Control Signals***

| Signal Name | Direction | Description |
|---|---|---|
| POR | Internal Control | Internal Power-On Reset (POR) circuit. |
| OSC | Internal Control | Internal configuration oscillator. |
| CRESET_B | Input | Configuration Reset input. Active-Low. No internal pull-up resistor. |
| CDONE[1] | Open-drain Output | Configuration Done output. Permanent, weak pull-up resistor to VCCIO_2. |

1. The iCE40-1KLP SWG16 package CDONE pin can be used as a user output. Please see iCE40 LP/HX Family Data Sheet for details.

The Power-On Reset circuit, POR, automatically resets the iCE40 component to a known state during power-up (cold boot). The POR circuit monitors the relevant voltage supply inputs, as shown in Figure 15-5. Once all supplies exceed their minimum thresholds, the configuration controller can start the configuration process.

The configuration controller begins configuring the iCE40 device, clocked by the Internal Oscillator, OSC. The OSC oscillator continues controlling configuration unless the iCE40 device is configured using the SPI Peripheral Configuration Interface.

***Figure 15-4. iCE40 Configuration Control Pins***

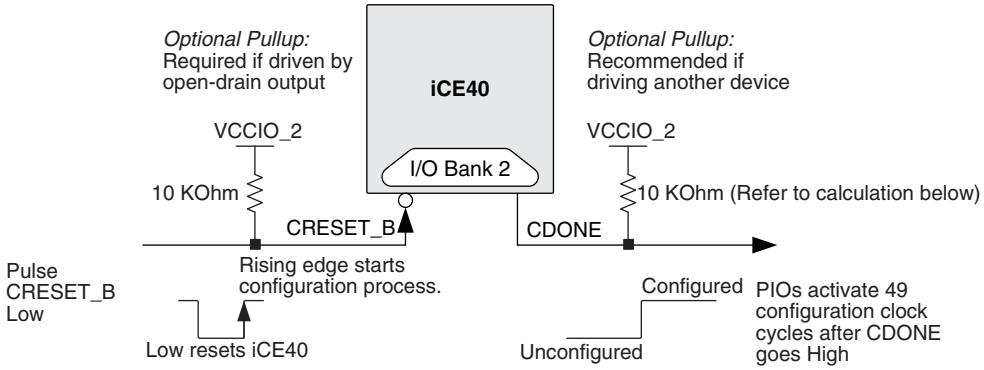



Figure 15-4 shows the two iCE40 configuration control pins, CRESET_B and CDONE. When driven Low, the dedicated Configuration Reset input, CRESET_B, resets the iCE40 device. When CRESET_B returns High, the iCE40 FPGA restarts the configuration process from its power-on conditions (Cold Boot). The CRESET_B pin is a pure input with no internal pull-up resistor. If driven by open-drain driver or un-driven, then connect the CRESET_B pin to a 10 KOhm pull-up resistor connected to the VCCIO_2 supply.

The iCE40 device signals the end of the configuration process by actively turning off the internal pull-down transistor on the Configuration Done output pin, CDONE. The pin has a permanent, weak internal pull-up resistor to the VCCIO_2 rail. However, depending on the system capacitance and configuration frequency, the CDONE pin must be tied to an external pullup resistor connected to the VCCIO_2 supply. The resistor size can be calculated knowing the configuration clock frequency (SCLK or MCLK) and the CDONE trace capacitance with the following formula:

*Rpullup=1/(2*ConfigFrequency*CDONETraceCap)*

The PIO pins activate according to their configured function after 49 configuration clock cycles. The internal oscillator is the configuration clock source for the SPI Master Configuration Interface and when configuring from Nonvolatile Configuration Memory (NVCM). When using the SPI Peripheral Configuration Interface, the configuration clock source is the SPI_SCK clock input pin.

## Internal Oscillator

During SPI Master or NVCM (iCE40 LP and iCE40 HX only) configuration mode, the controlling clock signal is generated from an internal oscillator. The oscillator starts operating at the default frequency. During the configuration process, however, bit settings within the configuration bitstream can specify a higher-frequency mode in order to maximize SPI bandwidth and reduce overall configuration time. See data sheet for the specified oscillator frequency range.

Using the SPI Master Configuration Interface, internal oscillator controls all the interface timing and clocks the SPI serial Flash PROM via the SPI_SCK clock output pin.

The oscillator output, which also supplies the SPI SCK clock output during the SPI Flash configuration process, has a 50% duty cycle.

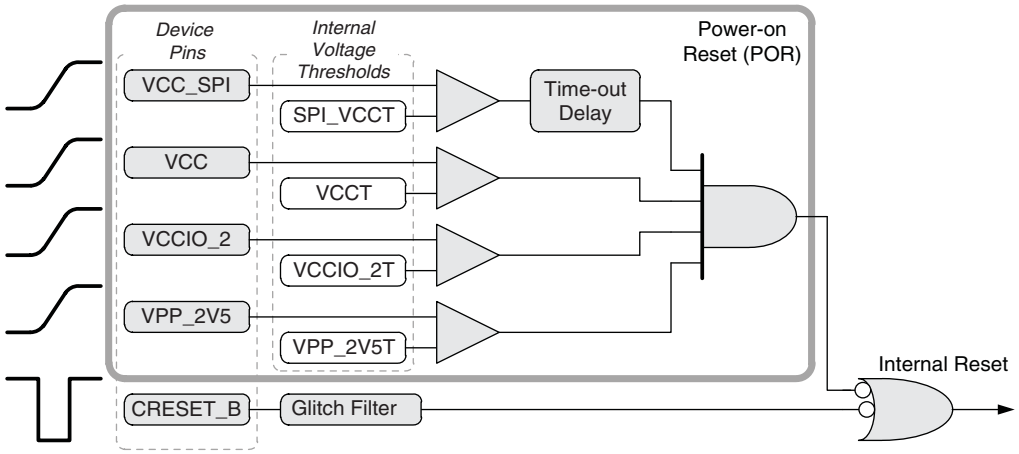
The Oscillator settings can be found in the iCEcube™ software by selecting the **Tools > Tool Options** pull down menu and then the **Bitstream** tab.

## Internal Device Reset

Figure 15-5 presents the various signals that internally reset the iCE40 internal logic.

- Power-On Reset (POR)
- CRESET_B Pin

***Figure 15-5. iCE40 Internal Reset Circuitry***




### Power-On Reset (POR)

The Power-on Reset (POR) circuit monitors specific voltage supply inputs and holds the device in reset until all the relevant supplies exceed the internal voltage thresholds. The VCC_SPI supply also has an additional time-out delay to allow an attached SPI serial PROM to power up properly. Table 15-4 shows the POR supply inputs. The Nonvolatile Configuration Memory (NVCM) requires that the VPP_2V5 supply be connected, even if the application does not use the NVCM.

Notes:

- It is recommend that Bank1 should be powered before the last supply gating POR.
- All banks must be powered prior to configuration.

***Table 15-4. Power-on Reset (POR) Voltage Resources***

| Supply Rail | iCE40 Production Devices |
|---|---|
| VCC | Yes |
| VCC_SPI | Yes |
| VCCIO_2 | Yes |
| VPP_2V5[1] | Yes |
| 1. Only needed for iCE40 LP and iCE40 HX devices, not for iCE40LM devices | |

### CRESET_B Pin

The CRESET_B pin resets the iCE40 internal logic when Low.

## sysCONFIG Port

The sysCONFIG port is used to program and configure the iCE40 FPGA. The device has a SPI configuration interface as the sysCONFIG port which can be used to configure the device.

***Table 15-5. sysCONFIG Ports***

| Interface | Port | Description |
|---|---|---|
| sysCONFIG | SPI Master Configuration interface | In this mode, the FPGA configures itself from an external SPI Flash or Diamond Programmer (version 2.2 or later). The FPGA behaves as master, generates internal clock and drives the clock to the external SPI Flash. |
| | SPI Slave Configuration interface | In this mode, the FPGA behaves as a Slave device. An external Application Processor, µC or Diamond Programmer (version 2.2 or later) configures or programs the device. |

## sysCONFIG Pins

The iCE40 FPGA has a set of sysCONFIG pins that are used to program and configure the device. The sysCONFIG pins are grouped together to create the sysCONFIG port, as discussed above. The sysCONFIG pins are dual-function, meaning they can be recovered as user I/O after configuration is complete. Table 15-6 shows the sysCONFIG pins.

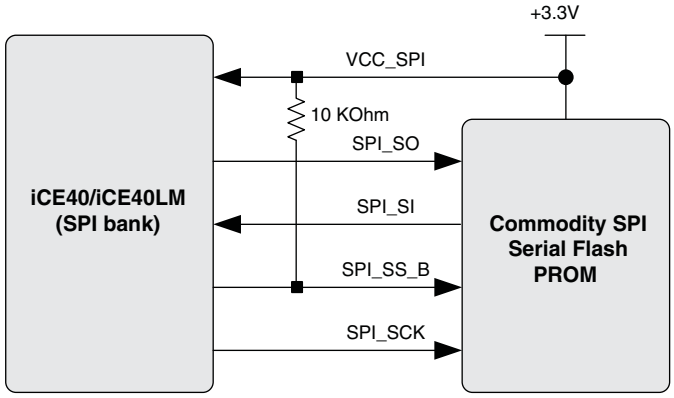
***Table 15-6. sysCONFIG Pins***

| Pin Name | Associated sysCONFIG Port | Pin Direction | Description |
|---|---|---|---|
| CRESET_B | — | Input | Configuration Reset input, active-low. No internal pull-up resistor. |
| CDONE | — | Output | Configuration Done output. The pin has a permanent, weak internal pull-up resistor to the VCCIO_2 rail. Depending on the frequency of configuration and the capacitance on CDONE node, then CDONE pin must be tied to an external pullup resistor connected to the VCCIO_2 supply. The resistor size can be calculated knowing the configuration clock frequency (SCLK or MCLK) and the CDONE trace capacitance with the following formula: *Rpullup=1/(2*ConfigFrequency*CDONETraceCap)* The iCE40-1KLP SWG16 package CDONE pin can be used as a user output. |
| SPI_SS_B | SPI Master/Slave configuration interface | Input/Output | An important dual-function, active-low slave select pin. After the device exits POR or CRESET_B is toggled (High-Low-High), it samples the SPI_SS_B to select the configuration mode (an output in Master mode and an input in Slave mode). iCE40LM devices have this pin shared with hardened SPI IP SPI_CSN pin. |
| SPI_SI | SPI Master/Slave configuration interface | Input | A dual-function, serial input pin in both configuration modes. iCE40LM devices have this pin shared with hardened SPI IP SPI_MOSI pin. |
| SPI_SO | SPI Master/Slave configuration interface | Output | A dual-function, serial output pin in both configuration modes. iCE40LM devices have this pin shared with hardened SPI IP SPI_MISO pin. |
| SPI_SCK | SPI Master/Slave configuration interface | Input/Output | A dual-function clock signal. An output in Master mode and input in Slave mode. iCE40LM devices have this pin shared with hardened SPI IP SPI_SCK pin. |

## SPI Master Configuration Interface

All iCE40 devices can be configured from an external, commodity SPI serial Flash PROM, as shown in Figure 15-6. The SPI configuration interface is essentially its own independent I/O bank, powered by the VCC_SPI supply input. Presently, most commercially-available SPI serial Flash PROMs require a 3.3V supply.

***Figure 15-6. iCE40 SPI Master Configuration Interface***




The SPI configuration interface is used primarily during development before mass production, where the configuration is then permanently programmed in the NVCM configuration memory (only available in iCE40 LP and iCE40 HX devices). However, the SPI interface can also be the primary configuration interface allowing easy in-system upgrades and support for multiple configuration images.

The SPI control signals are defined in Table 15-7.

***Table 15-7. SPI Master Configuration Interface Pins (SPI_SS_B High Before Configuration)***

| Signal Name | Direction | Description |
|---|---|---|
| VCC_SPI | Supply | SPI Flash PROM voltage supply input. |
| SPI_SO | Output | SPI Serial Output from the iCE40 device. |
| SPI_SI | Input | SPI Serial Input to the iCE40 device, driven by the select SPI serial Flash PROM. |
| SPI_SS_B | Output | SPI Slave Select output from the iCE40 device. Active Low. |
| SPI_SCK | Output | SPI Slave Clock output from the iCE40 device. |

After configuration, the SPI port pins are available to the user-application as additional PIO pins, supplied by the VCC_SPI input voltage.

### SPI PROM Requirements

The iCE40 SPI Flash configuration interface supports a variety of SPI Flash memory vendors and product families. However, Lattice does not specifically test, qualify, or otherwise endorse any specific SPI Flash vendor or product family. The iCE40 SPI interface supports SPI PROMs that they meet the following requirements.

- The PROM must operate at 3.3V or 1.8V in order to trigger the iCE40 FPGA's power-on reset circuit.
- The PROM must support the 0x0B Fast Read command, using a 24-bit start address and has 8 dummy bits before the PROM provides first data (see Figure 15-8).
- The PROM must have enough bits to program the iCE40 device.
- The PROM must support data operations at the upper frequency range for the selected iCE40 internal oscillator frequency (see data sheet). The oscillator frequency is selectable when creating the FPGA bitstream image.

- For lowest possible power consumption after configuration, the PROM should also support the 0xB9 Deep Power Down command and the 0xAB Release from Deep Power-down Command (see Figure 15-7 and Figure 15-9). The low-power mode is optional. The PROM must be powered and ready to be accessed following iCE40 POR
- The PROM must be ready to accept commands 10 µs after meeting its power-on conditions. In the PROM data sheet, this may be specified as $t_{VSL}$ or $t_{VCSL}$. It is possible to use slower PROMs by holding the CRESET_B input Low until the PROM is ready, then releasing CRESET_B, either under program control or using an external power-on reset circuit.

The iCEblink40™ development board and associated programming software uses an ST Micro/Numonyx M25Pxx SPI serial Flash PROM. Table 15-8 gives the bitstream sizes for different densities of the iCE40 FPGA that can be used to select a SPI Flash.

***Table 15-8. Bitstream Sizes for Different iCE40 FPGA Densities Used to Select a SPI Flash***

| Device | Bytes | Bits |
|---|---|---|
| iCE40-LP384 | 7872 | 62,976 |
| iCE40-LP/HX1K | 34,112 | 272,896 |
| iCE40-LP/HX4K | 136,448 | 1,091,584 |
| iCE40-LP/HX8K | 136,448 | 1,091,584 |
| iCE40LP640 | 34,112 | 272,896 |
| iCE40LM1K | 34,112 | 272,896 |
| iCE40LM2K | 68,224 | 545,792 |
| iCE40LM4K | 68,224 | 545,792 |

## Enabling SPI Configuration Interface

To enable the SPI configuration mode, the SPI_SS_B pin must be allowed to float High. The SPI_SS_B pin has an internal pull-up resistor. If SPI_SS_B is Low, then the iCE40 component defaults to the SPI Slave configuration mode.

## SPI Master Configuration Process

The iCE40 SPI Master Configuration Interface supports a variety of modern, high-density, low-cost SPI serial Flash PROMs. Most modern SPI PROMs include a power-saving Deep Power-down mode. The iCE40 component exploits this mode for additional system power savings.

The iCE40 SPI interface starts by driving SPI_SS_B Low, and then sends a Release from Power-down command to the SPI PROM, hexadecimal command code 0xAB. Figure 15-7 provides an example waveform. This initial command wakes up the SPI PROM if it is already in Deep Power-down mode. If the PROM is not in Deep Power-down mode, the extra command has no adverse affect other than that it requires a few additional microseconds during the configuration process. The iCE40 device transmits data on the SPI_SO output, on the falling edge of the SPI_SCK output. The SPI PROM does not provide any data to the iCE40 device's SPI_SI input. After sending the last command bit, the iCE40 device de-asserts SPI_SS_B High, completing the command. The iCE40 device then waits a minimum of 10 µS before sending the next SPI PROM command.

***Figure 15-7. SPI Release from Deep Power-down Command***

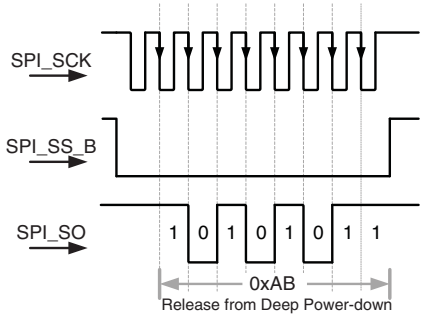
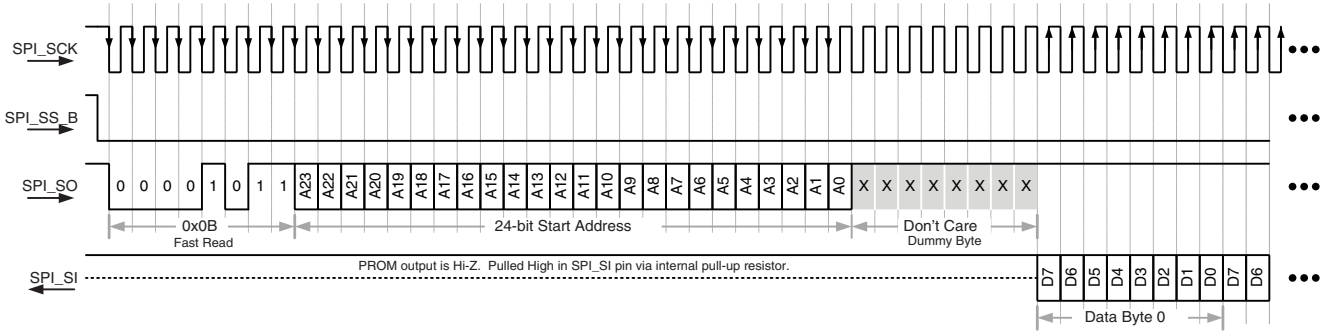



Figure 15-8 illustrates the next command issued by the iCE40 device. The iCE40 SPI interface again drives SPI_SS_B Low, followed by a Fast Read command, hexadecimal command code 0x0B, followed by a 24-bit start address, transmitted on the SPI_SO output. The iCE40 device provides data on the falling edge of SPI_SS_B. Upon initial power-up, the start address is always 0x00_0000. After waiting eight additional clock cycles, the iCE40 device begins reading serial data from the SPI PROM. Before presenting data, the SPI PROM's serial data output is high-impedance. The SPI_SI input pin has an internal pull-up resistor and sees high-impedance as logic '1'.

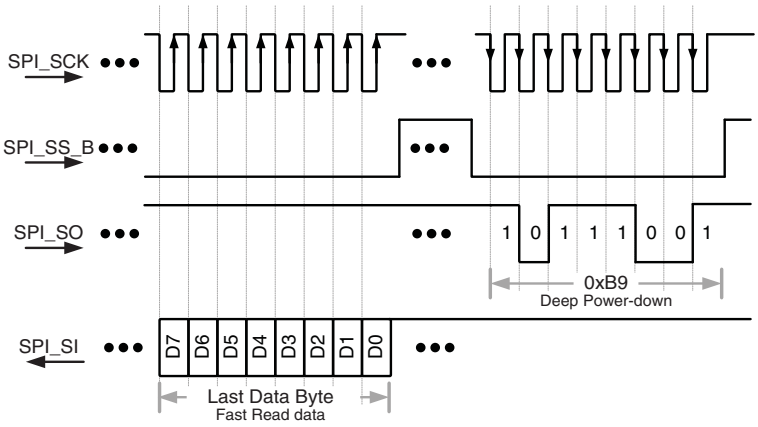
***Figure 15-8. SPI Fast Read Command***




The external SPI PROM supplies data on the falling edge of the iCE40 device's SPI_SCK clock output. The iCE40 device captures each PROM data value on the SPI_SI input, using the rising edge of the SPI_SCK clock signal. The SPI PROM data starts at the 24-bit address presented by the iCE40 device. PROM data is serially output, byte by byte, with most-significant bit, D7, presented first. The PROM automatically increments an internal byte counter as long as the PROM is selected and clocked.

After transferring the required number configuration data bits, the iCE40 device ends the Fast Read command by de-asserting its SPI_SS_B PROM select output, as shown in Figure 15-9. To conserve power, the iCE40 device then optionally issues a final Deep Power-down command, hexadecimal command code 0xB9. After de-asserting the SPI_SS_B output, the SPI PROM enters its Deep Power-down mode. The final power-down step is optional; the application may wish to use the SPI PROM and can skip this step, controlled by a configuration option.

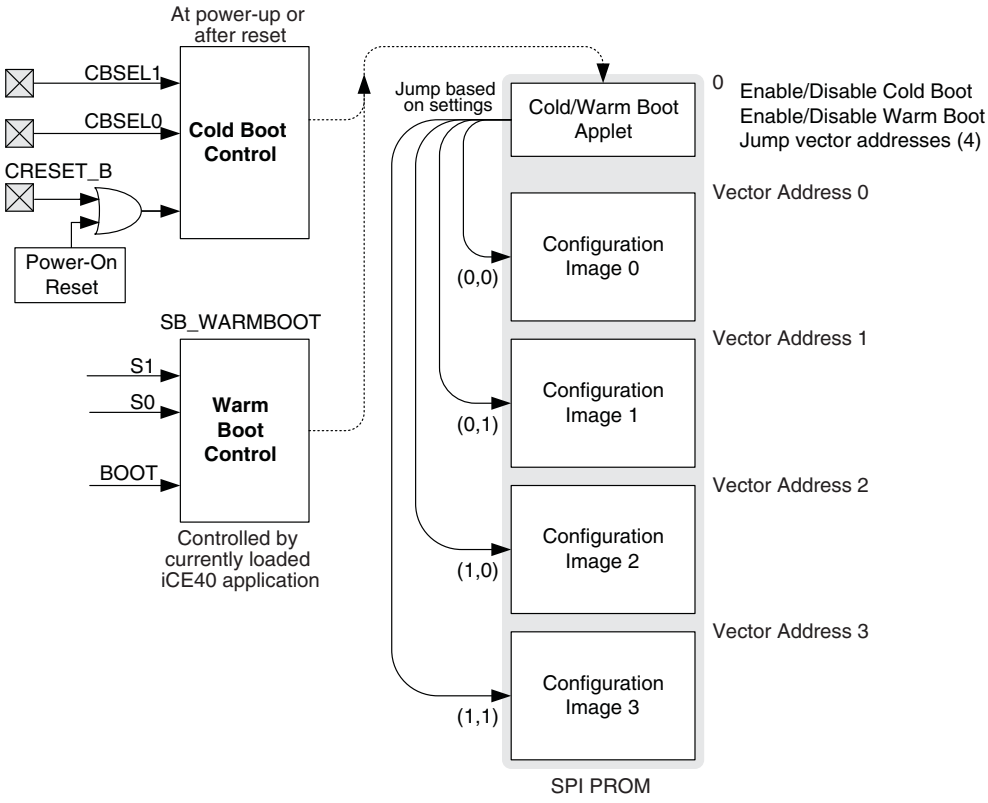
*Figure 15-9. Final Configuration Data, SPI Deep Power-down Command*



## Cold Boot Configuration Option *(Applies to iCE40 LP and iCE40 HX Devices Only)*

By default, the iCE40 FPGA is programmed with a single configuration image, either from internal NVCM memory, from an external SPI Flash PROM, or externally from a processor or microcontroller.

***Figure 15-10. Cold Boot and Warm Boot Configuration***




When self-loading from an SPI Flash PROM, the FPGA supports an additional configuration option called Cold Boot mode. When this option is enabled in the configuration bitstream, the iCE40 FPGA boots normally from power-on or a master reset (CRESET_B = Low pulse), but monitors the value on two PIO pins that are borrowed during configuration, as shown in Figure 15-10. These pins, labeled PIO2/CBSEL0 and PIO2/CBSEL1, tell the FPGA which of the four possible SPI configurations to load into the device.

- Load from initial location, either from NVCM or from address 0 in SPI Flash PROM. For Cold Boot or Warm Boot applications, the initial configuration image contains the cold boot/warm boot applet.
- Check if Cold Boot configuration feature is enabled in the bitstream.
  - If not enabled, FPGA configures normally.
  - If Cold Boot is enabled, then the FPGA reads the logic values on pins CBSEL[1:0]. The FPGA uses the value as a vector and then reads from the indicated vector address.
  - At the selected CBSEL[1:0] vector address, there is a starting address for the selected configuration image.
    - For SPI Flash PROMs, the new address is a 24-bit start address in Flash.
- Using the new start address, the FPGA restarts reading configuration memory from the new location.

When creating the initial configuration image, the Lattice development software loads the start address for up to four configuration images in the bitstream. The value on the CBSEL[1:0] pins tells the configuration controller to

read a specific start address, then to load the configuration image stored at the selected address. The multiple bitstreams are stored in the SPI Flash.

After configuration, the CBSEL[1:0] pins become normal PIO pins available to the application.

The Cold Boot feature allows the iCE40 to be reprogrammed for special application requirements such as the following.

- A normal operating mode and a self-test or diagnostics mode.
- Different applications based on switch settings.
- Different applications based on a card-slot ID number. Use external SPI Flash PROMs only.

## Warm Boot Configuration Option *(Applies to iCE40 LP and iCE40 HX Devices Only)*

The Warm Boot configuration is similar to the Cold Boot feature, but is completely under the control of the FPGA application.

A special design primitive, SB_WARMBOOT, allows an FPGA application to choose between four configuration images using two internal signal ports, S1 and S0, as shown in Figure 15-10. These internal signal ports connect to programmable interconnect, which in turn can connect to PLB logic and/or PIO pins.

After selecting the desired configuration image, the application then asserts the internal signal BOOT port High to force the FPGA to restart the configuration process from the specified vector address stored in PROM.

## Time-Out and Retry

When configuring from external SPI Flash, the iCE40 device looks for a synchronization word. If the device does not find a synchronization word within its timeout period, the device automatically attempts to restart the configuration process from the very beginning. This feature is designed to address any potential power-sequencing issues that may occur between the iCE40 device and the external PROM.

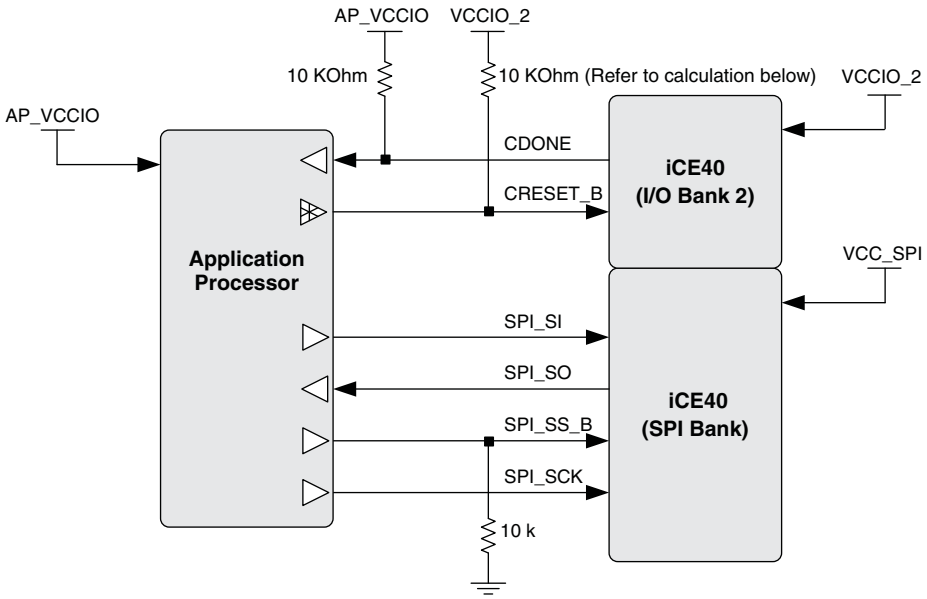
The iCE40 device attempts to reconfigure six times. If not successful after six attempts, the iCE40 FPGA automatically goes into low-power mode.

## SPI Slave Configuration Interface

Using the SPI slave configuration interface, an application processor (AP) serially writes a configuration image to an iCE40 FPGA using the iCE40's SPI interface, as shown in Figure 15-6. The iCE40's SPI configuration interface is a separate, independent I/O bank, powered by the VCC_SPI supply input. Typically, VCC_SPI is the same voltage as the application processor's I/O. The configuration control signals, CDONE and CRESET_B, are supplied by the separate I/O Bank 2 voltage input, VCCIO_2.

This same SPI slave interface supports the programming of the Nonvolatile Configuration Memory (NVCM) of the iCE40 LP and iCE40 HX.

*Figure 15-11. iCE40 SPI Slave Configuration Interface*




The SPI control signals are defined in Table 15-9.

*Table 15-9. SPI Slave Configuration Interface Pins (SPI_SS_B Low when CRESET_B Released)*

| Signal Name | Direction | iCE40 I/O Supply | Description |
|---|---|---|---|
| CDONE | Output | VCCIO_2 | Configuration Done output from iCE40. Connect to a external pull-up resistor to the application processor I/O voltage, AP_VCC. The resistor size can be calculated knowing the configuration clock frequency (SCLK or MCLK) and the CDONE trace capacitance with the following formula: *Rpullup=1/(2*ConfigFrequency*CDONETraceCap)* The iCE40-1KLP SWG16 package CDONE pin can be used as a user output. |
| CRESET_B | Input | | Configuration Reset input on iCE40. Typically driven by AP using an open-drain driver, which also requires a 10 KOhm pull-up resistor to VCCIO_2. |
| VCC_SPI | Supply | VCC_SPI | SPI Flash PROM voltage supply input. |
| SPI_SI | Input | | SPI Serial Input to the iCE40 FPGA, driven by the application processor. |
| SPI_SO | Output | | SPI Serial Output from CE65 device to the application processor. Not actually used during SPI peripheral mode configuration but required if the SPI interface is also used to program the NVCM. |
| SPI_SS_B | Input | | SPI Slave Select output from the application processor. Active Low. Optionally hold Low prior to configuration using a 10 KOhm pull-down resistor to ground. |
| SPI_SCK | Input | | SPI Slave Clock output from the application processor. |

After configuration, the SPI port pins are available to the user-application as additional PIO pins, supplied by the VCC_SPI input voltage.

## Enabling SPI Configuration Interface

The optional 10 KOhm pull-down resistor on the SPI_SS_B signal ensures that the iCE40 FPGA powers up in the SPI peripheral mode. Optionally, the application processor drives the SPI_SS_B pin Low when CRESET_B is released, forcing the iCE40 FPGA into SPI peripheral mode.

## SPI Slave Configuration Process

Figure 15-12 illustrates the interface timing for the SPI slave mode and Figure 15-13 outlines the resulting configuration process. The actual timing specifications appear in the data sheet. The application processor (AP) begins by driving the iCE40 CRESET_B pin Low, resetting the iCE40 FPGA. Similarly, the AP holds the iCE40's SPI_SS_B pin Low. The AP must hold the CRESET_B pin Low for at least 200 ns. Ultimately, the AP either releases the CRESET_B pin and allows it to float High via the 10 KOhm pull-up resistor to VCCIO_2 or drives CRESET_B High. The iCE40 FPGA enters SPI peripheral mode when the CRESET_B pin returns High while the SPI_SS_B pin is Low,

After driving CRESET_B High or allowing it to float High, the AP must wait a minimum of 300 µs, allowing the iCE40 FPGA to clear its internal configuration memory.

After waiting for the configuration memory to clear, the AP sends the configuration image generated by the Diamond Programmer (version 2.2 or later). An SPI slave mode configuration image must not use the Cold Boot or Warm Boot options. Send the entire configuration image, without interruption, serially to the iCE40's SPI_SI input on the falling edge of the SPI_SCK clock input. Once the AP sends the 0x7EAA997E synchronization pattern, the generated SPI_SCK clock frequency must be within the data sheet specified range while sending the configuration image. Send each byte of the configuration image with most-significant bit (msb) first. The AP sends data to the iCE40 FPGA on the falling edge of the SPI_SCK clock. The iCE40 FPGA internally captures each incoming SPI_SI data bit on the rising edge of the SPI_SCK clock. The SPI_SO output pin in the iCE40 LP and iCE40 HX is not used during SPI slave mode but must connect to the AP if the AP also programs the Nonvolatile Configuration Memory (NVCM) of the iCE40 LP and iCE40 HX.

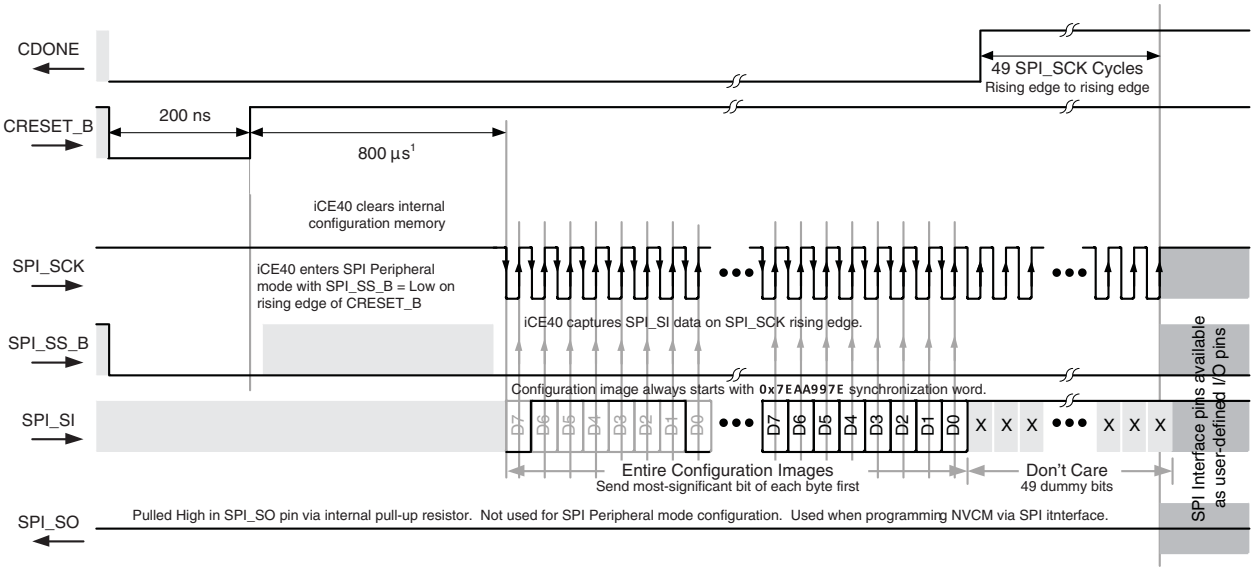
After sending the entire image, the iCE40 FPGA releases the CDONE output allowing it to float High via the external pull-up resistor to AP_VCC. If the CDONE pin remains Low, then an error occurred during configuration and the AP should handle the error accordingly for the application.

After the CDONE output pin goes High, send at least 49 additional dummy bits, effectively 49 additional SPI_SCK clock cycles measured from rising-edge to rising-edge.

After the additional SPI_CLK cycles, the SPI interface pins then become available to the user application loaded in FPGA. In the iCE40-1KLP SWG16 package, the CDONE pin can be used as a user output.

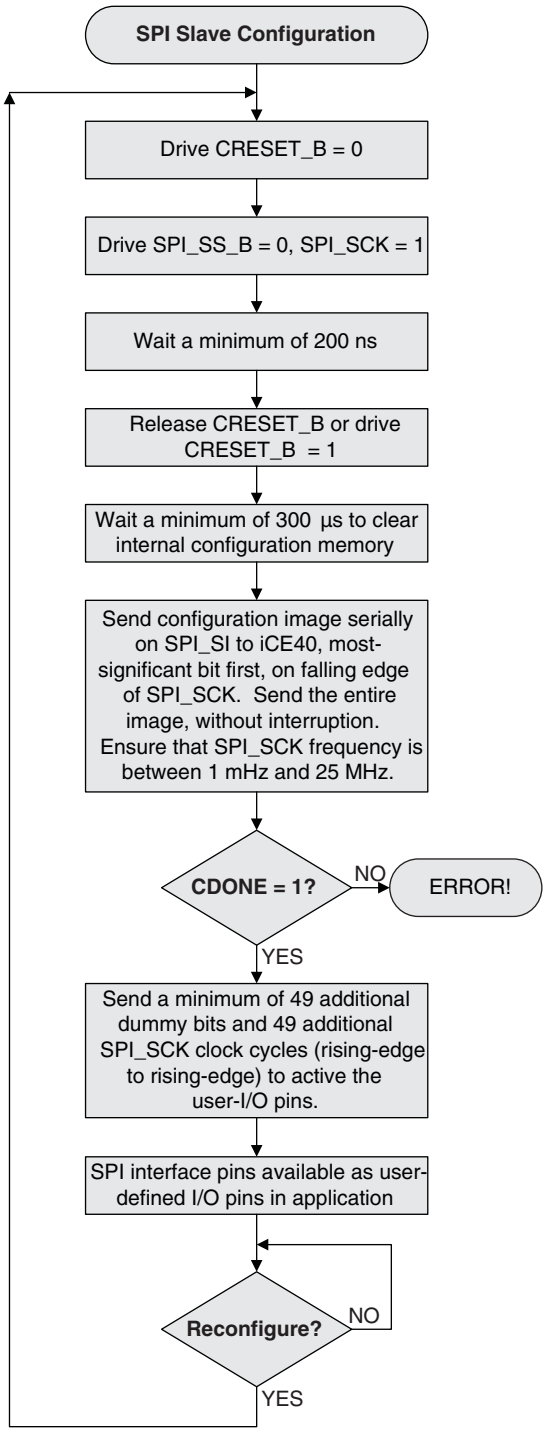
To reconfigure the iCE40 FPGA or to load a different configuration image, merely restart the configuration process by pulsing CRESET_B Low or power-cycling the FPGA.

*Figure 15-12. Application Processor Waveforms for SPI Peripheral Mode Configuration Process*




1. Refer to Appendix A for timing based on density.

*Figure 15-13. SPI Slave Configuration Process*




Refer to Appendix A for the SPI peripheral configuration procedure.

## Voltage Compatibility

As shown in Figure 15-6, there are potentially three different supply voltages involved in the SPI Peripheral interface, described in Table 15-10.

***Table 15-10. SPI Peripheral Mode Supply Voltages***

| Supply Voltage | Description |
|---|---|
| AP_VCCIO | I/O supply to the Application Processor (AP) |
| VCC_SPI | Voltage supply for the iCE40 SPI interface |
| VCCIO_2 | Supply voltage for the iCE40 I/O Bank 2 |

Table 15-11 describes how to maintain voltage compatibility for two interface scenarios. The easiest interface is when the Application Processor's (AP) I/O supply rail and the iCE40's SPI and VCCIO_2 bank supply rails all connect to the same voltage. The second scenario is when the AP's I/O supply voltage is greater than the iCE40's VCCIO_2 supply voltage.

***Table 15-11. CRESET_B and CDONE Voltage Compatibility***

| Condition | CRESET_B | | | CDONE Pull-up | Requirement |
|---|---|---|---|---|---|
| | Direct | Open-Drain | Pull-up | | |
| AP_VCCIO = VCC_SPI<br>AP_VCCIO = VCCIO_2 | OK | OK with pull-up | Required if using open-drain output | Recommended | AP can directly drive CRESET_B High and Low although an open-drain output recommended is if multiple devices control CRESET_B. If using an open-drain driver, the CRESET_B input must include a 10 KOhm pull-up resistor to VCCIO_2. The 10 KOhm pull-up resistor to AP_VCCIO is also recommended. |
| AP_VCCIO > VCCIO_2 | N/A | Required, requires pull-up | Required | Required | The AP must control CRESET_B with an open-drain output, which requires a 10 KOhm pull-up resistor to VCCIO_2. The 10 KOhm pull-up resistor to AP_VCCIO is required. |

# Technical Support Assistance

e-mail: techsupport@latticesemi.com

Internet: www.latticesemi.com

## Revision History

| Date | Version | Change Summary |
|---|---|---|
| March 2012 | 01.3 | Initial release. |
| June 2012 | 01.4 | Updated document with new corporate style. |
| September 2012 | 01.5 | Updated based on latest iCE40 information:<br>- Included configuration and programming information<br>- Updated bit file sizes<br>- Updated Table 3. Power-on Reset (POR) Voltage Resources<br>- Removed JTAG references<br>- Include configuration algorithm in Appendix A |
| February 2013 | 01.6 | Updated the Bitstream Sizes for Different iCE40 FPGA Densities Used to Select a SPI Flash table. |
| April 2013 | 01.7 | Updated the Pseudo Code for configuring an iCE40 device. |
| | | Added information on how to access the Oscillator settings. |
| | | Defined iCE40LP/HX4K wait time to be 1200us. |
| | 01.8 | Updated the SPI Peripheral Configuration Process section. |
| June 2013 | 01.9 | Updated the Cold Boot Configuration Option section to provide more accurate information regarding the Cold Boot mode and the storage of multiple bitstreams. |
| | | Changed VCCIO_AP to AP_VCCIO in Table 5-10. |
| July 2013 | 02.0 | Updated SPI PROM Requirements section. |
| | | Updated Technical support Assistance information. |
| August 2013 | 02.1 | Added information regarding resistor size calculation in the Configuration Control Signals section; updated the iCE40 Configuration Control Pins figure. |
| | | Added notes in the Power-On Reset (POR) section. |
| | | Updated description of CDONE pin in the sysCONFIG Pin and SPI Peripheral Configuration Interface Pins (SPI_SS_B Low when CRESET_B Released tables. |
| | | Updated the iCE40 SPI Master Configuration Interface figure. |
| | | Updated CDONE information in the SPI Configuration Process section, |
| October 2013 | 02.2 | Updated for iCE40LM support. |

# Appendix A. SPI Slave Configuration Procedure

## CPU Configuration Procedure

The sequence for configuring the iCE40 SRAM follows.

***Table 15-12. iCE40 SRAM Configuration Sequence***

| Index | Step | Action | Description |
|---|---|---|---|
| 1 | Power up the iCE40 FPGA or toggle its CRESET signal low for 200ns and toggle back to high with SPI_SS_B = 0, forcing the device to enter Slave mode. Keep SPI_SS_B low until step 4. | Hold SPI_SS_B =0 and toggle CRESET | See Figure 15-14 below. |
| 2 | Wait > = 800us to 1200us until the iCE40 FPGA completes internal housekeeping work and is ready to receive CPU bitmap data and instructions. | Wait >= 1200us for iCE40LP/HX4K, iCE40LP/HX8K | |
| 3 | From this point on, the iCE40 FPGA requires CPU provides operation clock through the SPI_SCK pin of the iCE40 FPGA until configuration is complete. | Feed clock to the SPI_CLK pin | |
| 4 | Read and start sending the FPGA bitmap to the iCE40 device. Data goes to the SPI_SDI pin. | Tx bitmap data from user memory | Each clock shifts one bit of data at the clock falling edge. Hold previous SPI_SS_B state, no toggle.<br>Complete sending all bits in bitmap file. Important: Continuous clock is required. |
| 5 | Wait for 100 clocks. | Shifting 100 clocks | Configuration complete after 100 clocks. |
| 6 | Monitor the CDONE pin. It should go to high. Otherwise, the configuration fails and stops. | Check CDONE Hi | Device in operation when CDONE = 1; device fails when CDONE = 0. |

## Configuration Waveforms

### iCE40 Reset Waveforms

The reset timing waveforms for initiating NVCM programming are shown below.

***Figure 15-14. iCE40 Reset Waveform 1***

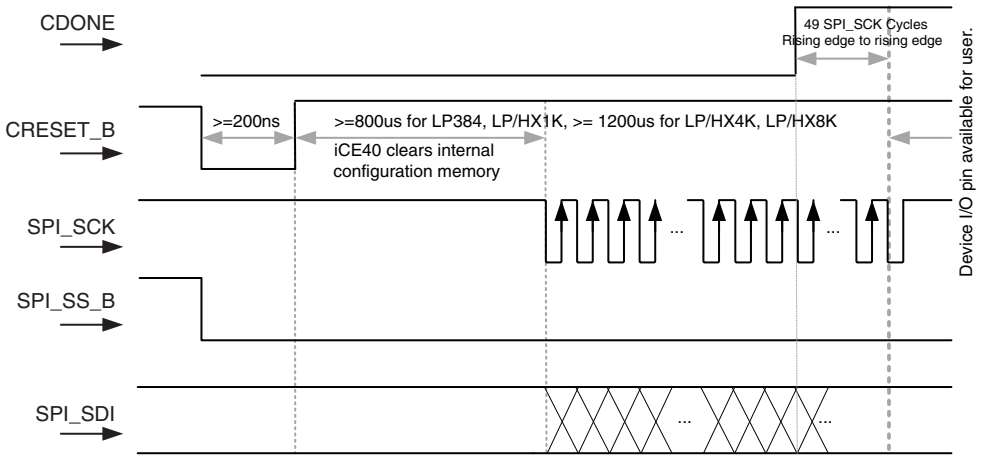


*Figure 15-15. iCE40 Timing Waveform 2*

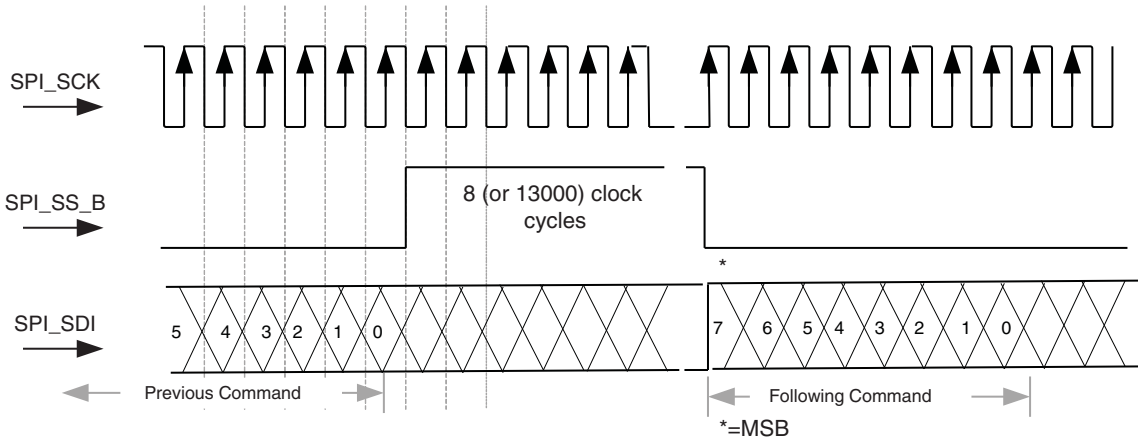



*Figure 15-16. iCE40 Timing Waveform 3*

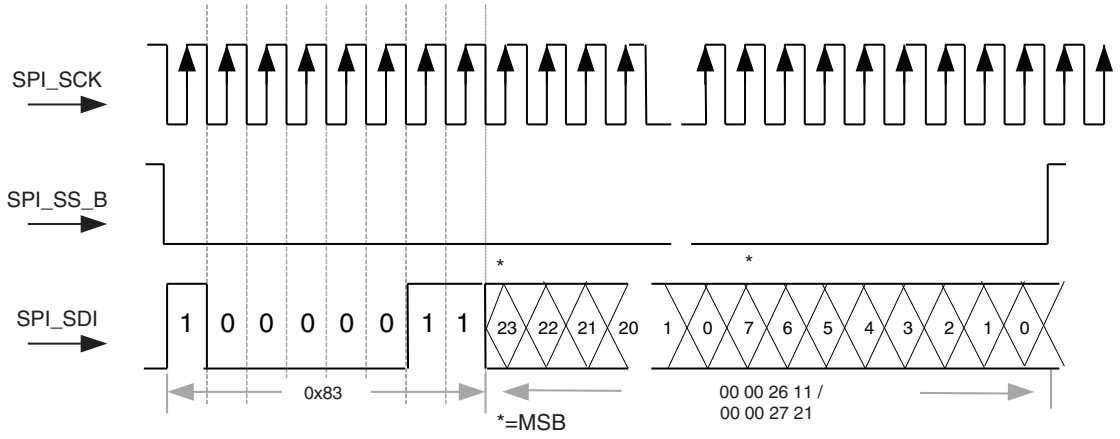
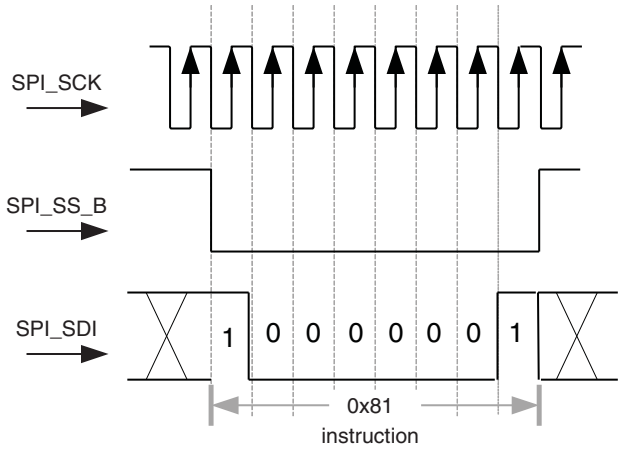



*Figure 15-17. iCE40 Timing Waveform 4*



## Pseudo Code

### Configuration

The pseudo code included below will configure an iCE40 device. It assumes a raw binary file generated from iCEcube will be used (*.bin). Alternatively, a hex file is also generated from iCEcube and can be used as well. The implementation should only make the necessary text-to-binary conversions.

```
//
//  iCE40 Configuration Pseudo-code
//
void Config_iCE40 (type, file)
{
   //
   // Open Hex File early to avoid clock delay later
   //
   file_pointer = fopen(file);// Open bin file

   //
   // Reset the iCE40 Device
   //
   Set_Port(SPI_SS_B, false);// Set SPI_SS_B low
   Set_Port(CRESET, false);// Set CRESET low
   Set_Port(SPI_CLK, true);// Set SPI_CLK high
   nSec_Delay(200);// Delay 200 nsec
   Set_Port(CRESET, true);// Set CRESET high
   if (type == L1K or L4K)
     uSec_Delay(800);// Delay 800 usec if L1K,L4K
   else if (type == L8K)
     uSec_Delay(1200);// Delay 1200 usec for L8K
   Set_Port(SPI_SS_B, true);// Set SPI_SS_B high
   Send_Clocks (8);// Send 8 clocks
   //
   // Send data from bin file
   //
   Send_File(file_pointer);// Send bin file
   Send_Clocks (100);// Send 100 clocks
   //
   // Verify successful configuration
   //
   if (Get_Port(CDONE))
     Return PASS;// PASS if CDONE is true
   else
     Return FAIL;// FAIL if CDONE is false
}

//
//  Clock Generation 10MHz
//
void Send_Clocks(num_clocks)
{
   for {i = 0; i < num_clocks; i++}
   {
     Set_Port(SPI_CLOCK, false);// Set SPI_CLK low
     nSec_Delay(50);// Delay 50 nsec
```

```
      Set_Port(SPI_CLOCK, true);// Set SPI_CLK high
      nSec_Delay(50);// Delay 50 nsec
   }
}

//
//  Send Data from file
//
void Send_File(file_pointer)
{
   byte = getc(file_pointer);// Read first byte from file
   while (byte != EOF)
   {
      Send_Byte (byte);// Send data byte
      byte = getc(file_pointer);// read next byte from file
   }
}
```

# Memory Usage Guide for iCE40 Devices

October 2013 Technical Note TN1250

## Introduction

This technical note discusses memory usage for the iCE40™ device family. It is intended to be used as a guide to the high-speed synchronous RAM Blocks and the iCE40 sysMEM™ Embedded Block RAM (EBR). The EBR is the embedded block RAM of the device, each 4Kbit in size.

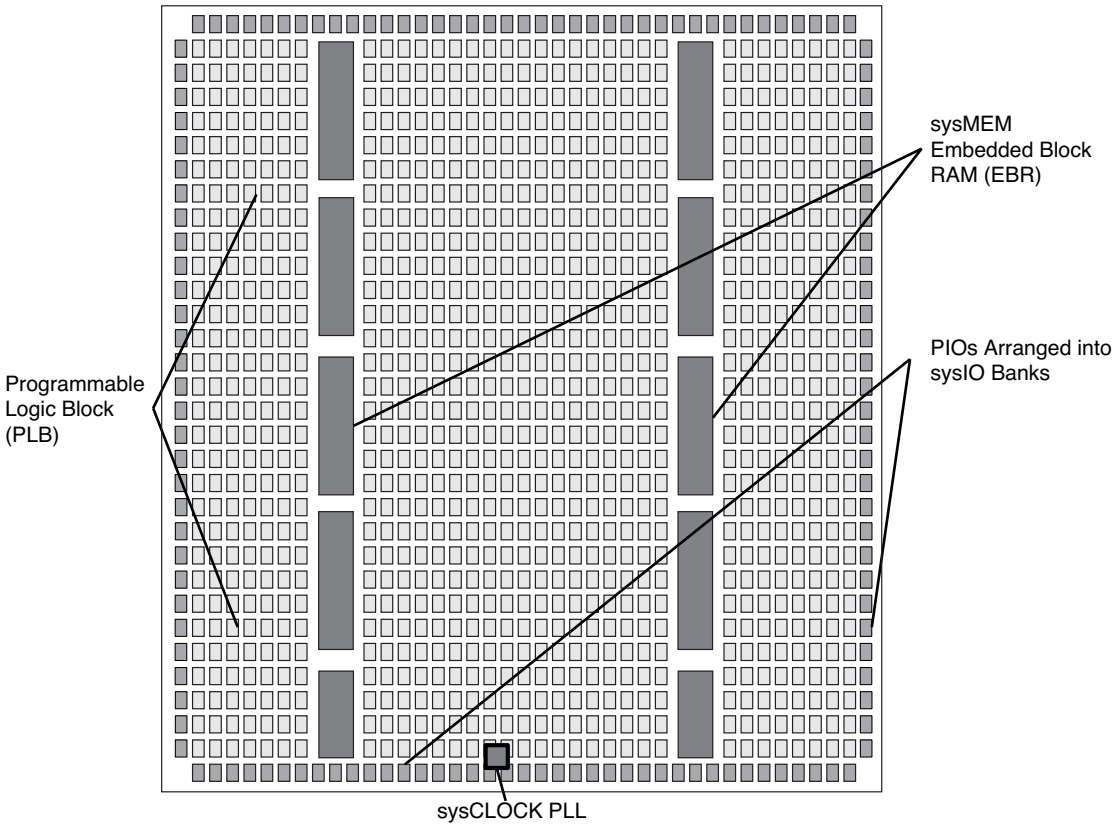
The iCE40 device architecture provides resources for memory-intensive applications. Single-Port RAM, Dual-Port RAM and FIFO can be constructed using the EBRs. The EBRs can be utilized by instantiating software primitives as described later in this document. Apart from primitive instantiation, the iCECube2™ design software infers generic codes as EBRs.

## Memories in iCE40 Devices

iCE40 devices contain an array of EBRs.

Figure 16-1 shows the placement of EBRs in a typical iCE40 device (does not represent true numbers of design elements).

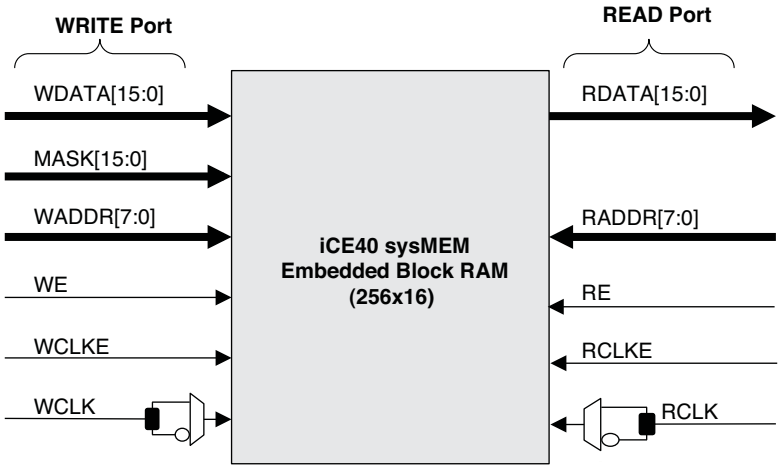
***Figure 16-1. Typical Layout of an iCE40 Device***



## iCE40 sysMEM Embedded Block RAM

Each iCE40 device includes multiple high-speed synchronous EBRs, each 4Kbit in size. A single iCE40 device integrates between eight and 32 such blocks. Each EBR is a 256-word deep by 16-bit wide, two-port register file, as illustrated in Figure 16-2. The input and output connections to and from an EBR feed into the programmable interconnect resources.

***Figure 16-2. sysMEM Embedded Block RAM***




Using programmable logic resources, an EBR implements a variety of logic functions, each with configurable input and output data widths.

- Random-access memory (RAM)
  - Single-port RAM with a common address, enable, and clock control lines
  - Two-port RAM with separate read and write control lines, address inputs, and enable
- Register file and scratchpad RAM
- First-In, First-Out (FIFO) memory for data buffering applications
- 256-deep by 16-wide ROM with registered outputs; contents loaded during configuration
- Counters, sequencers

As shown in Figure 16-2, an EBR has separate write and read ports, each with independent control signals. Table 16-1 lists the signals for both ports. Additionally, the write port has an active-low bit-line write-enable control; optionally mask write operations on individual bits. By default, input and output data is 16 bits wide, although the data width is configurable using programmable logic and, if needed, multiple EBRs.

The WCLK and RCLK inputs optionally connect to one of the following clock sources:

- The output from any one of the eight Global Buffers, or
- A connection from the general-purpose interconnect fabric

## Signals

Table 16-1 lists the signal names, direction, and function of each connection to the EBR block.

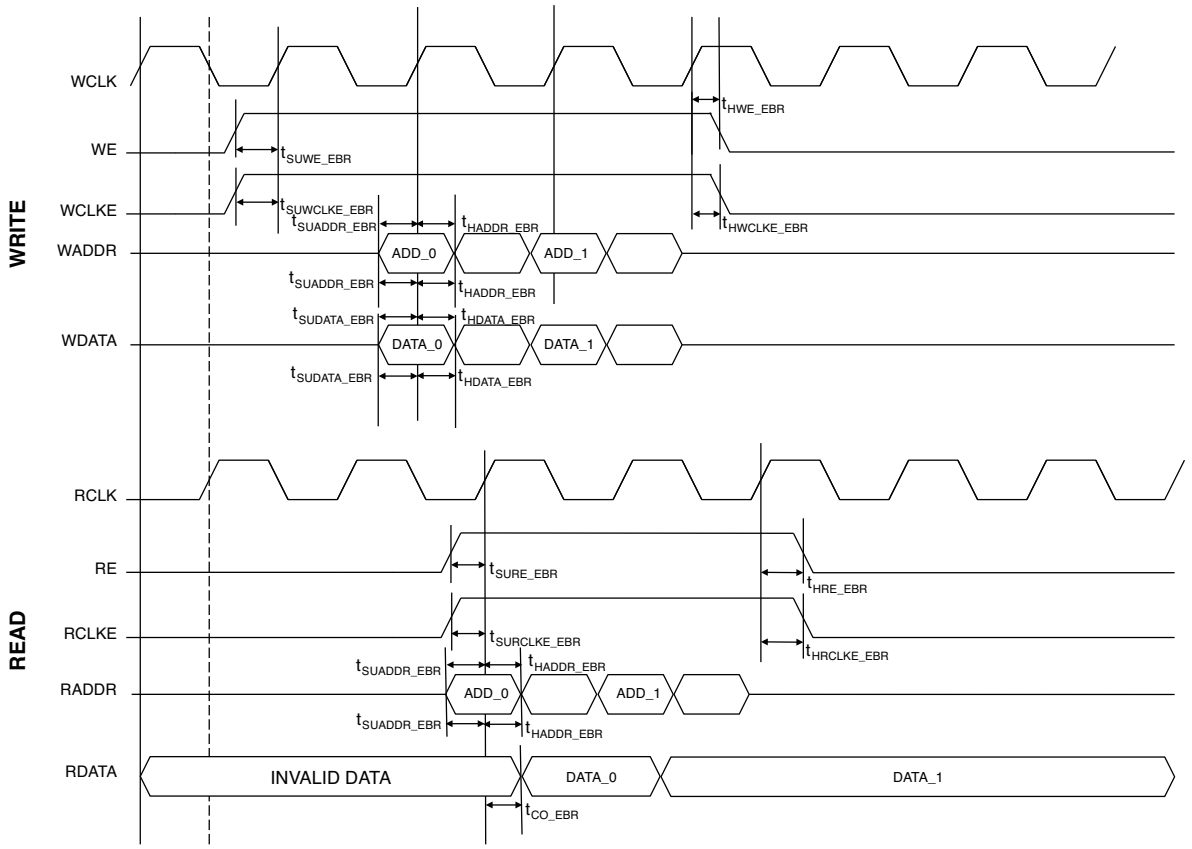
***Table 16-1. EBR Signal Descriptions***

| Signal Name | Direction | Description |
|---|---|---|
| WDATA[15:0] | Input | Write Data input. |
| MASK[15:0] | Input | Masks write operations for individual data bit-lines.<br>0 = write bit; 1 = don't write bit |
| WADDR[7:0] | Input | Write Address input. Selects one of 256 possible RAM locations. |
| WE | Input | Write Enable input. |
| WCLKE | Input | Write Clock Enable input. |
| WCLK | Input | Write Clock input. Default rising-edge, but with falling-edge option. |
| RDATA[15:0] | Output | Read Data output. |
| RADDR[7:0] | Input | Read Address input. Selects one of 256 possible RAM locations. |
| RE | Input | Read Enable input. |
| RCLKE | Input | Read Clock Enable input. |
| RCLK | Input | Read Clock input. Default rising-edge, but with falling-edge option. |

## Timing Diagram

Figure 16-3 shows the timing diagram for the EBR memory module.

***Figure 16-3. EBR Module Timing Diagram[1]***




1. Internal timing values are considered in the iCEcube2 software's place and route.

## Write Operations

By default, all EBR write operations are synchronized to the rising edge of WCLK although the clock is invertible as shown in Figure 16-2.When the WCLKE signal is low, the clock to the EBR block is disabled, keeping the EBR in its lowest power mode.

To write data into the EBR block, perform the following operations:

- Supply a valid address on the WADDR[7:0] address input port
- Supply valid data on the WDATA[15:0] data input port
- To write or mask selected data bits, set the associated MASK input port accordingly. For example, write operations on data bit D[i] are controlled by the associated MASK[i] input.
  - MASK[i] = 0: Write operations are enabled for data line WDATA[i]
  - MASK[i] = 1: Mask write operations are disabled for data line WDATA[i]
- Enable the EBR write port (WE = 1)
- Enable the EBR write clock (WCLKE = 1)
- Apply a rising clock edge on WCLK (assuming that the clock is not inverted)

## Read Operations

By default, all EBR read operations are synchronized to the rising edge of RCLK although the clock is invertible as shown in Figure 16-2.

To read data from the EBR block, perform the following operations:

- Supply a valid address on the RADDR[7:0] address input port
- Enable the EBR read port (RE = 1)
- Enable the EBR read clock (RCLKE = 1)
- Apply a rising clock edge on RCLK
- After the clock edge, the EBR contents located at the specified address (RADDR) appear on the RDATA output port

## EBR Considerations

### Read Data Register Undefined Immediately After Configuration

Unlike the flip-flops in the Programmable Logic Blocks and Programmable I/O pins, the RDATA port is not automatically reset after configuration. Consequently, immediately following configuration and before the first valid Read Data operation, the initial RDATA read value is undefined.

### Pre-loading EBR Data

The data contents for an EBR block can be optionally pre-loaded during iCE40 configuration. If not pre-loaded during configuration, then the EBR contents must be initialized by the iCE40 application before the EBR contents are valid. EBR initialization data can be done in the RTL code. Pre-loading the EBR data in the configuration bitstream increases the size of the configuration image accordingly.

### EBR Contents Preserved During Configuration

EBR contents are preserved (write protected) during configuration, assuming that voltage supplies are maintained throughout. Consequently, data can be passed between multiple iCE40 configurations by leaving it in an EBR block and then skipping pre-loading during the subsequent reconfiguration.

**Low-Power Setting**

To place an EBR block in its lowest power mode, keep WCLKE = 0 and RCLKE = 0. In other words, when not actively using an EBR block, disable the clock inputs.

## iCE40 sysMEM Embedded Block RAM Memory Primitives

This section lists the iCE40 sysMEM EBR software primitives that can be instantiated in the RTL. Different EBRs are used in different configurations. Each EBR has separate write and read ports, each with independent control signals. Each EBR can be configured into a RAM block of size 256x16, 512x8, 1024x4 or 2048x2. The data contents of the EBR can optionally be pre-loaded during iCE40 device configuration by specifying the initialization data in the primitive instantiation.

Table 16-2 lists the supported dual port synchronous RAM configurations, each of 4Kbits in size. The RAM blocks can be directly instantiated in the top module and taken through the iCube2 software flow.

***Table 16-2. EBR Configurations and Primitive Names***

| Block RAM Configuration | Block RAM Configuration and Size | WADDR Port Size (Bits) | WDATA Port Size (Bits) | RADDR Port Size (Bits) | RDATA Port Size (Bits) | MASK Port Size (Bits) |
|---|---|---|---|---|---|---|
| SB_RAM256x16<br>SB_RAM256x16NR<br>SB_RAM256x16NW<br>SB_RAM256x16NRNW | 256x16 (4K) | 8 [7:0] | 16 [15:0] | 8 [7:0] | 16 [15:0] | 16 [15:0] |
| SB_RAM512x8<br>SB_RAM512x8NR<br>SB_RAM512x8NW<br>SB_RAM512x8NRNW | 512x8 (4K) | 9 [8:0] | 8 [7:0] | 9 [8:0] | 8 [7:0] | No Mask Port |
| SB_RAM1024x4<br>SB_RAM1024x4NR<br>SB_RAM1024x4NW<br>SB_RAM1024x4NRNW | 1024x4 (4K) | 10 [9:0] | 4 [3:0] | 10 [9:0] | 4 [3:0] | No Mask Port |
| SB_RAM2048x2<br>SB_RAM2048x2NR<br>SB_RAM2048x2NW<br>SB_RAM2048x2NRNW | 2048x2 (4K) | 11 [10:0] | 2 [1:0] | 11 [10:0] | 2 [1:0] | No Mask Port |

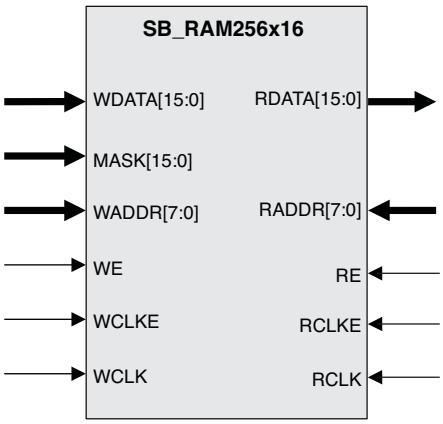
For iCE40 EBR primitives with a negative-edged read or write clock, the base primitive name is appended with a ‘N’ and a ‘R’ or ‘W’ depending on the clock that is affected (see Table 16-3 for the 256x16 RAM block configuration).

***Table 16-3. Naming Convention for RAM Primitives***

| RAM Primitive Name | Description |
|---|---|
| SB_RAM256x16 | Positive-edged read clock, positive-edged write clock |
| SB_RAM256x16NR | Negative-edged read clock, positive-edged write clock |
| SB_RAM256x16NW | Positive-edged read clock, negative-edged write clock |
| SB_RAM256x16NRNW | Negative-edged read clock, negative-edged write clock |

## SB_RAM256x16

*Figure 16-4. SB_RAM256x16 Primitive*




### Verilog Instantiation

```
SB_RAM256x16 ram256X16_inst (
    .RDATA(RDATA_c[15:0]),
    .RADDR(RADDR_c[7:0]),
    .RCLK(RCLK_c),
    .RCLKE(RCLKE_c),
    .RE(RE_c),
    .WADDR(WADDR_c[7:0]),
    .WCLK(WCLK_c),
    .WCLKE(WCLKE_c),
    .WDATA(WDATA_c[15:0]),
    .WE(WE_c),
    .MASK(MASK_c[15:0])
);

defparam ram256x16_inst.INIT_0 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram256x16_inst.INIT_1 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram256x16_inst.INIT_2 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram256x16_inst.INIT_3 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram256x16_inst.INIT_4 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram256x16_inst.INIT_5 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram256x16_inst.INIT_6 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram256x16_inst.INIT_7 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram256x16_inst.INIT_8 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram256x16_inst.INIT_9 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram256x16_inst.INIT_A =
256'h0000000000000000000000000000000000000000000000000000000000000000;
```

```
defparam ram256x16_inst.INIT_B =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram256x16_inst.INIT_C =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram256x16_inst.INIT_D =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram256x16_inst.INIT_E =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram256x16_inst.INIT_F =
256'h0000000000000000000000000000000000000000000000000000000000000000;
```

**VHDL Instantiation**

```
ram256X16_inst : SB_RAM256x16
generic map (
INIT_0 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_1 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_2 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_3 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_4 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_5 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_6 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_8 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_9 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_A => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_B => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_C => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_D => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_E => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_F => X"0000000000000000000000000000000000000000000000000000000000000000"
)

port map (
   RDATA => RDATA_c,
   RADDR => RADDR_c,
   RCLK => RCLK_c,
   RCLKE => RCLKE_c,
   RE => RE_c,
   WADDR => WADDR_c,
   WCLK=> WCLK_c,
   WCLKE => WCLKE_c,
   WDATA => WDATA_c,
   MASK => MASK_c,
   WE => WE_c
);
```

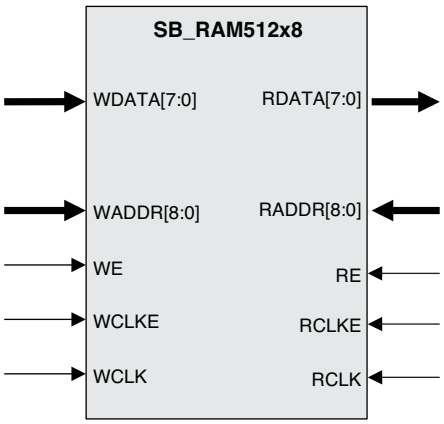
Table 16-4 is a complete list of SB_RAM256x16 based primitives.

*Table 16-4. SB_RAM256x16 Based Primitives*

| Primitive | Description |
|---|---|
| SB_RAM256x16 | SB_RAM256x16 //Positive edged clock RCLK WCLK<br>(RDATA, RCLK, RCLKE, RE, RADDR, WCLK, WCLKE, WE, WADDR, MASK, WDATA); |
| SB_RAM256x16NR | SB_RAM256x16NR // Negative edged Read Clock – i.e. RCLKN<br>(RDATA, RCLKN, RCLKE, RE, RADDR, WCLK, WCLKE, WE, WADDR, MASK, WDATA); |
| SB_RAM256x16NW | SB_RAM256x16NW // Negative edged Write Clock – i.e. WCLKN<br>(RDATA, RCLK, RCLKE, RE, RADDR, WCLKN, WCLKE, WE, WADDR, MASK, WDATA); |
| SB_RAM256x16NRNW | SB_RAM256x16NRNW // Negative edged Read and Write – i.e. RCLKN WRCKLN<br>(RDATA, RCLKN, RCLKE, RE, RADDR, WCLKN, WCLKE, WE, WADDR, MASK, WDATA); |

## SB_RAM512x8

*Figure 16-5. SB_RAM512x8 Primitive*




### Verilog Instantiation

```
SB_RAM512x8 ram512X8_inst (
    .RDATA(RDATA_c[7:0]),
    .RADDR(RADDR_c[8:0]),
    .RCLK(RCLK_c),
    .RCLKE(RCLKE_c),
    .RE(RE_c),
    .WADDR(WADDR_c[8:0]),
    .WCLK(WCLK_c),
    .WCLKE(WCLKE_c),
    .WDATA(WDATA_c[7:0]),
    .WE(WE_c)
);

defparam ram512x8_inst.INIT_0 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram512x8_inst.INIT_1 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram512x8_inst.INIT_2 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram512x8_inst.INIT_3 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram512x8_inst.INIT_4 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
```

```
defparam ram512x8_inst.INIT_5 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram512x8_inst.INIT_6 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram512x8_inst.INIT_7 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram512x8_inst.INIT_8 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram512x8_inst.INIT_9 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram512x8_inst.INIT_A =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram512x8_inst.INIT_B =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram512x8_inst.INIT_C =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram512x8_inst.INIT_D =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram512x8_inst.INIT_E =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram512x8_inst.INIT_F =
256'h0000000000000000000000000000000000000000000000000000000000000000;
```

**VHDL Instantiation**

```
ram512X8_inst : SB_RAM512x8
generic map (
INIT_0 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_1 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_2 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_3 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_4 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_5 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_6 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_8 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_9 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_A => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_B => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_C => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_D => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_E => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_F => X"0000000000000000000000000000000000000000000000000000000000000000"
)

port map (
   RDATA => RDATA_c,
   RADDR => RADDR_c,
   RCLK => RCLK_c,
   RCLKE => RCLKE_c,
   RE => RE_c,
   WADDR => WADDR_c,
   WCLK=> WCLK_c,
   WCLKE => WCLKE_c,
   WDATA => WDATA_c,
```

```
   WE => WE_c
);
WE => WE_c
);
```

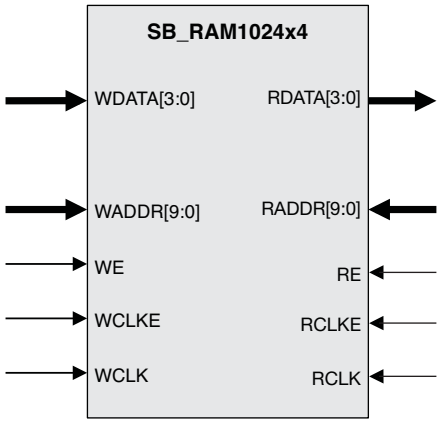
Table 16-5 is a complete list of SB_RAM512x8 based primitives.

***Table 16-5. SB_RAM512x8 Based Primitives***

| Primitive | Description |
|---|---|
| SB_RAM512x8 | SB_RAM512x8 //Positive edged clock RCLK WCLK<br>(RDATA, RCLK, RCLKE, RE, RADDR, WCLK, WCLKE, WE, WADDR, MASK, WDATA); |
| SB_RAM512x8NR | SB_RAM512x8NR // Negative edged Read Clock – i.e. RCLKN<br>(RDATA, RCLKN, RCLKE, RE, RADDR, WCLK, WCLKE, WE, WADDR, MASK, WDATA); |
| SB_RAM512x8NW | SB_RAM512x8NW // Negative edged Write Clock – i.e. WCLKN<br>(RDATA, RCLK, RCLKE, RE, RADDR, WCLKN, WCLKE, WE, WADDR, MASK, WDATA); |
| SB_RAM512x8NRNW | SB_RAM512x8NRNW // Negative edged Read and Write – i.e. RCLKN WRCKLN<br>(RDATA, RCLKN, RCLKE, RE, RADDR, WCLKN, WCLKE, WE, WADDR, MASK, WDATA); |

## SB_RAM1024x4

***Figure 16-6. SB_RAM1024x4 Primitive***




**Verilog Instantiation**

```
SB_RAM1024x4 ram1024x4_inst (
   .RDATA(RDATA_c[3:0]),
   .RADDR(RADDR_c[9:0]),
   .RCLK(RCLK_c),
   .RCLKE(RCLKE_c),
   .RE(RE_c),
   .WADDR(WADDR_c[3:0]),
   .WCLK(WCLK_c),
   .WCLKE(WCLKE_c),
   .WDATA(WDATA_c[9:0]),
   .WE(WE_c)
);

defparam ram1024x4_inst.INIT_0 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram1024x4_inst.INIT_1 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
```

```
defparam ram1024x4_inst.INIT_2 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram1024x4_inst.INIT_3 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram1024x4_inst.INIT_4 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram1024x4_inst.INIT_5 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram1024x4_inst.INIT_6 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram1024x4_inst.INIT_7 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram1024x4_inst.INIT_8 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram1024x4_inst.INIT_9 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram1024x4_inst.INIT_A =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram1024x4_inst.INIT_B =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram1024x4_inst.INIT_C =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram1024x4_inst.INIT_D =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram1024x4_inst.INIT_E =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram1024x4_inst.INIT_F =
256'h0000000000000000000000000000000000000000000000000000000000000000;
```

**VHDL Instantiation**

```
Ram1024X4_inst : SB_RAM1024x4
generic map (
INIT_0 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_1 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_2 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_3 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_4 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_5 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_6 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_8 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_9 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_A => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_B => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_C => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_D => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_E => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_F => X"0000000000000000000000000000000000000000000000000000000000000000"
)

port map (
RDATA => RDATA_c,
RADDR => RADDR_c,
RCLK => RCLK_c,
```

```
RCLKE => RCLKE_c,
RE => RE_c,
WADDR => WADDR_c,
WCLK=> WCLK_c,
WCLKE => WCLKE_c,
WDATA => WDATA_c,
WE => WE_c
);
```

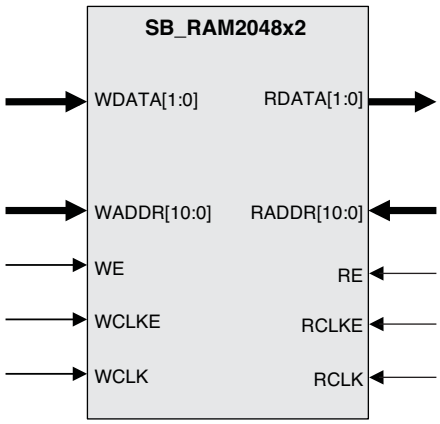
Table 16-6 is a complete list of SB_RAM1024x4 based primitives.

***Table 16-6. SB_RAM1024x4 Based Primitives***

| Primitive | Description |
|---|---|
| SB_RAM1024x4 | SB_RAM1024x4 //Positive edged clock RCLK WCLK<br>(RDATA, RCLK, RCLKE, RE, RADDR, WCLK, WCLKE, WE, WADDR, MASK, WDATA); |
| SB_RAM1024x4NR | SB_RAM1024x4NR // Negative edged Read Clock – i.e. RCLKN<br>(RDATA, RCLKN, RCLKE, RE, RADDR, WCLK, WCLKE, WE, WADDR, MASK, WDATA); |
| SB_RAM1024x4NW | SB_RAM1024x4NW // Negative edged Write Clock – i.e. WCLKN<br>(RDATA, RCLK, RCLKE, RE, RADDR, WCLKN, WCLKE, WE, WADDR, MASK, WDATA); |
| SB_RAM1024x4NRNW | SB_RAM1024x4NRNW // Negative edged Read and Write – i.e. RCLKN WRCKLN<br>(RDATA, RCLKN, RCLKE, RE, RADDR, WCLKN, WCLKE, WE, WADDR, MASK, WDATA); |

## SB_RAM2048x2

***Figure 16-7. SB_RAM2048x2***




**Verilog Instantiation**

```
SB_RAM2048x2 ram2048x2_inst (
    .RDATA(RDATA_c[2:0]),
    .RADDR(RADDR_c[10:0]),
    .RCLK(RCLK_c),
    .RCLKE(RCLKE_c),
    .RE(RE_c),
    .WADDR(WADDR_c[2:0]),
    .WCLK(WCLK_c),
    .WCLKE(WCLKE_c),
    .WDATA(WDATA_c[10:0]),
    .WE(WE_c)
);
```

```
defparam ram2048x2_inst.INIT_0 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram2048x2_inst .INIT_1 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram2048x2_inst .INIT_2 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram2048x2_inst .INIT_3 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram2048x2_inst .INIT_4 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram2048x2_inst .INIT_5 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram2048x2_inst .INIT_6 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram2048x2_inst .INIT_7 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram2048x2_inst .INIT_8 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram2048x2_inst .INIT_9 =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram2048x2_inst .INIT_A =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram2048x2_inst .INIT_B =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram2048x2_inst .INIT_C =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram2048x2_inst .INIT_D =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram2048x2_inst .INIT_E =
256'h0000000000000000000000000000000000000000000000000000000000000000;
defparam ram2048x2_inst .INIT_F =
256'h0000000000000000000000000000000000000000000000000000000000000000;
```

**VHDL Instantiation**

```
Ram2048x2_inst : SB_RAM2048x2
generic map (
INIT_0 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_1 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_2 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_3 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_4 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_5 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_6 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_7 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_8 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_9 => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_A => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_B => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_C => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_D => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_E => X"0000000000000000000000000000000000000000000000000000000000000000",
INIT_F => X"0000000000000000000000000000000000000000000000000000000000000000"
)
```

```
port map (
   RDATA => RDATA_c,
   RADDR => RADDR_c,
   RCLK => RCLK_c,
   RCLKE => RCLKE_c,
   RE => RE_c,
   WADDR => WADDR_c,
   WCLK=> WCLK_c,
   WCLKE => WCLKE_c,
   WDATA => WDATA_c,
   WE => WE_c
);
```

Table 16-7 is a complete list of the SB_RAM2048x2 based primitives.

***Table 16-7. SB_RAM2048x2 Based Primitives***

| Primitive | Description |
|---|---|
| SB_RAM2048x2 | SB_RAM2048x2 //Positive edged clock RCLK WCLK<br>(RDATA, RCLK, RCLKE, RE, RADDR, WCLK, WCLKE, WE, WADDR, MASK, WDATA); |
| SB_RAM2048x2NR | SB_RAM2048x2NR // Negative edged Read Clock – i.e. RCLKN<br>(RDATA, RCLKN, RCLKE, RE, RADDR, WCLK, WCLKE, WE, WADDR, MASK, WDATA); |
| SB_RAM2048x2NW | SB_RAM2048x2NW // Negative edged Write Clock – i.e. WCLKN<br>(RDATA, RCLK, RCLKE, RE, RADDR, WCLKN, WCLKE, WE, WADDR, MASK, WDATA); |
| SB_RAM2048x2NRNW | SB_RAM2048x2NRNW // Negative edged Read and Write – i.e. RCLKN WRCKLN<br>(RDATA, RCLKN, RCLKE, RE, RADDR, WCLKN, WCLKE, WE, WADDR, MASK, WDATA); |

## SB_RAM40_4K

SB_RAM40_4K is the basic physical RAM primitive which can be instantiated and configured to different depths and data ports. The SB_RAM40_4K block has a size of 4 Kbits with separate write and read ports, each with independent control signals. By default, input and output data is 16 bits wide, although the data width is configurable using the READ_MODE and WRITE_MODE parameters. The data contents of the SB_RAM40_4K block are optionally pre-loaded during iCE device configuration.

***Table 16-8. SB_RAM40_4K Naming Convention Rules***

| RAM Primitive Name | Description |
|---|---|
| SB_RAM40_4K | Positive-edged read clock, positive-edged write clock |
| SB_RAM40_4KNR | Negative-edged read clock, positive-edged write clock |
| SB_RAM40_4KNW | Positive-edged read clock, negative-edged write clock |
| SB_RAM40_4KNRNW | Negative-edged clock, negative-edged write clock |

*Figure 16-8. SB_RAM40_4K*

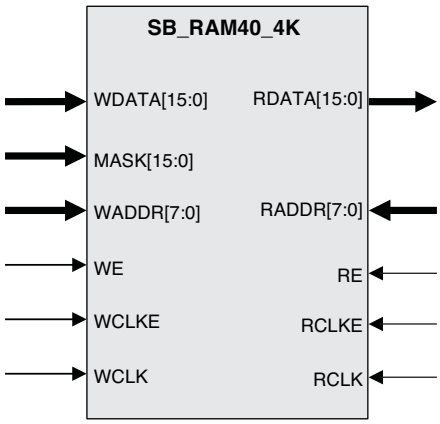



Table 16-9 lists the signals for both ports.

*Table 16-9. SB_RAM40_4K Signal Descriptions*

| Signal Name | Direction | Description |
|---|---|---|
| WDATA[15:0] | Input | Write Data input. |
| MASK[15:0] | Input | Bit-line Write Enable input, active low. Applicable only when WRITE_MODE parameter is set to '0'. |
| WADDR[7:0] | Input | Write Address input. Selects up to 256 possible locations. |
| WE | Input | Write Enable input, active high. |
| WCLK | Input | Write Clock input, rising-edge active. |
| WCLKE | Input | Write Clock Enable input. |
| RDATA[15:0] | Output | Read Data output. |
| RADDR[7:0] | Input | Read Address input. Selects one of 256 possible locations. |
| RE | Input | Read Enable input, active high. |
| RCLK | Input | Read Clock input, rising-edge active. |
| RCLKE | Input | Read Clock Enable input. |

Table 16-10 describes the parameter values to infer the desired RAM configuration.

*Table 16-10. SB_RAM40_4K Primitive Parameter Descriptions*

| Parameter Name | Description | Parameter Value | Configuration |
|---|---|---|---|
| INIT_0, … …,INIT_F | RAM Initialization Data. Passed using 16 parameter strings, each comprising 256 bits. (16x256=4096 total bits) | INIT_0 to INIT_F | Initialize the RAM with predefined value |
| WRITE_MODE | Sets the RAM block write port configuration | 0 | 256x16 |
| | | 1 | 512x18 |
| | | 2 | 1024x4 |
| | | 3 | 2048x2 |
| READ_MODE | | 0 | 256x16 |
| | | 1 | 512x8 |
| | | 2 | 1024x4 |
| | | 3 | 2048x2 |

**Verilog Instantiation**

```
// Physical RAM Instance without Pre Initialization
SB_RAM40_4K ram40_4kinst_physical (
    .RDATA(RDATA),
    .RADDR(RADDR),
    .WADDR(WADDR),
    .MASK(MASK),
    .WDATA(WDATA)
    .RCLKE(RCLKE),
    .RCLK(RCLK),
    .RE(RE),
    .WCLKE(WCLKE),
    .WCLK(WCLK),
    .WE(WE)
);

defparam ram40_4kinst_physical.READ_MODE=0;
defparam ram40_4kinst_physical.WRITE_MODE=0;
```

**VHDL Instantiation**

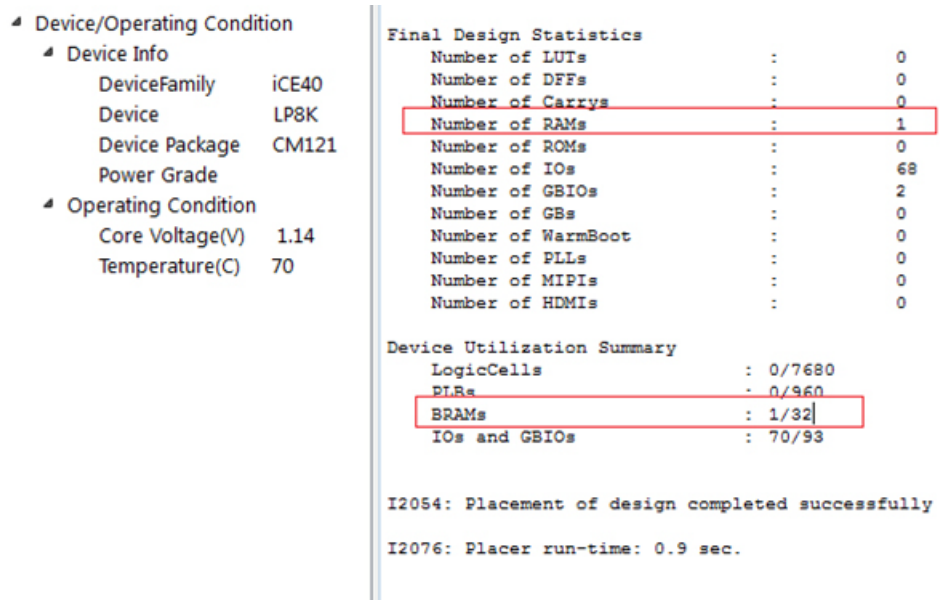
```
-- Physical RAM Instance without Pre Initialization
ram40_4kinst_physical : SB_RAM40_4K
generic map (
    READ_MODE => 0,
    WRITE_MODE= >0
)

port map (
    RDATA=>RDATA,
    RADDR=>RADDR,
    WADDR=>WADDR,
    MASK=>MASK,
    WDATA=>WDATA,
    RCLKE=>RCLKE,
    RCLK=>RCLK,
    RE=>RE,
    WCLKE=>WCLKE,
    WCLK=>WCLK,
    WE=>WE
);
```

## EBR Utilization Summary in iCEcube2 Design Software

The placer.log file in the iCEcube2 design software shows the device utilization summary. The Final Design Statistics and Device Utilization Summary sections show the number of EBRs (or RAMs) used against the total number. Figure 16-9 shows the EBR usage when one SB_RAM256x16 was instantiated in the design.

*Figure 16-9. iCEcube2 Design Software Report File*




## Technical Support Assistance

e-mail: techsupport@latticesemi.com

Internet: www.latticesemi.com

## Revision History

| Date | Version | Change Summary |
|---|---|---|
| September 2012 | 01.0 | Initial release. |
| October 2013 | 01.1 | Removed iCE40 Family EBRs table. |
| | | Updated Technical Support Assistance information. |

# Appendix A. Standard HDL Code References

This appendix contains standard HDL (Verilog and VHDL) codes for popular memory elements, which can be used to infer a sysMEM EBR automatically. Standard HDL coding techniques do not require you to know the details of the Block RAMs of the device and are inferred automatically.

## Single-Port RAM

### Verilog

```
module ram (din, addr, write_en, clk, dout);// 512x8
  parameter addr_width = 9;
  parameter data_width = 8;
  input [addr_width-1:0] addr;
  input [data_width-1:0] din;
  input write_en, clk;
  output [data_width-1:0] dout;
  reg [data_width-1:0] dout; // Register for output.
  reg [data_width-1:0] mem [(1<<addr_width)-1:0];

  always @(posedge clk)
  begin
    if (write_en)
      mem[(addr)] <= din;
    dout = mem[addr]; // Output register controlled by clock.
  end
endmodule
```

### VHDL

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity ram is
generic (
  addr_width : natural := 9;--512x8
  data_width : natural := 8);
port (
  addr : in  std_logic_vector (addr_width - 1 downto 0);
  write_en : in  std_logic;
  clk : in  std_logic;
  din : in  std_logic_vector (data_width - 1 downto 0);
  dout : out std_logic_vector (data_width - 1 downto 0));
end ram;

architecture rtl of ram is
  type mem_type is array ((2** addr_width) - 1 downto 0) of
    std_logic_vector(data_width - 1 downto 0);
  signal mem : mem_type;
begin
  process (clk)
  begin
    if (clk'event and clk = '1') then
      if (write_en = '1') then
        mem(conv_integer(addr)) <= din;
      end if;
```

```
            dout <= mem(conv_integer(addr));
    end if;

      -- Output register controlled by clock.
  end process;
end rtl;
```

## Dual Port Ram

### Verilog

```
module ram (din, write_en, waddr, wclk, raddr, rclk, dout);//512x8
  parameter addr_width = 9;
  parameter data_width = 8;
  input [addr_width-1:0] waddr, raddr;
  input [data_width-1:0] din;
  input write_en, wclk, rclk;
  output reg [data_width-1:0] dout;
  reg [data_width-1:0] mem [(1<<addr_width)-1:0]
     ;

  always @(posedge wclk) // Write memory.
  begin
    if (write_en)
      mem[waddr] <= din; // Using write address bus.
  end
  always @(posedge rclk) // Read memory.
  begin
    dout <= mem[raddr]; // Using read address bus.
  end
endmodule
```

### VHDL

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity ram is
generic (
  addr_width : natural := 9;--512x8
  data_width : natural := 8);
port (
  write_en : in  std_logic;
  waddr : in  std_logic_vector (addr_width - 1 downto 0);
  wclk : in  std_logic;
  raddr : in  std_logic_vector (addr_width - 1 downto 0);
  rclk : in  std_logic;
  din : in  std_logic_vector (data_width - 1 downto 0);
  dout : out std_logic_vector (data_width - 1 downto 0));
end ram;

architecture rtl of ram is
  type mem_type is array ((2** addr_width) - 1 downto 0) of
    std_logic_vector(data_width - 1 downto 0);
  signal mem : mem_type;
```

```
begin
  process (wclk)
  -- Write memory.
  begin
    if (wclk'event and wclk = '1') then
      if (write_en = '1') then
        mem(conv_integer(waddr)) <= din;
          -- Using write address bus.
      end if;
    end if;
  end process;
  process (rclk) -- Read memory.
  begin
    if (rclk'event and rclk = '1') then
      dout <= mem(conv_integer(raddr));
        -- Using read address bus.
    end if;
  end process;
end rtl;
```

# iCE40 sysCLOCK PLL Design and Usage Guide

October 2013 Technical Note TN1251

## Introduction

This technical note discusses the clock resources available in the iCE40™ devices (iCE40LP/HX, iCE40LM). Details are provided for global buffers and sysCLOCK™ PLLs. The iCE40 devices include an ultra-low power Phase Locked Loop (PLL) to support a variety of display, imaging and memory interface applications. Table 17-1 lists the number of sysCLOCK PLLs and global buffers in the iCE40 device family.

***Table 17-1. Number of PLLs in the iCE40 Device Family[1]***

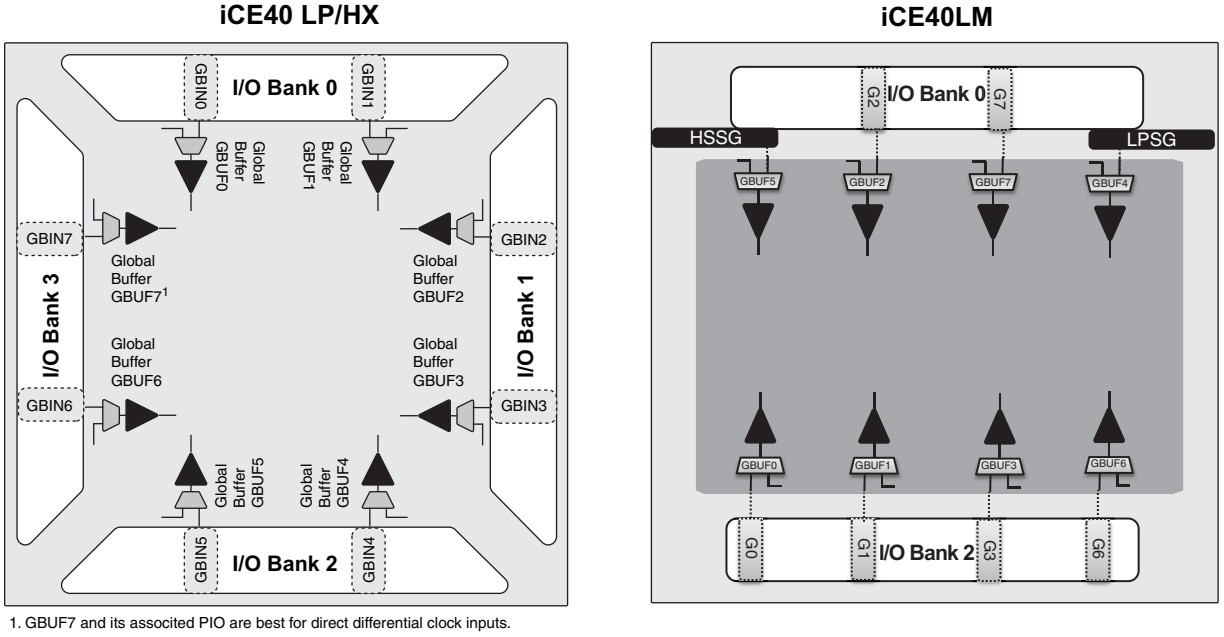
| Parameter | LP384 | LP1K | LP4K | LP8K | HX1K | HX4K | HX8K | LM1K | LM2K | LM4K |
|---|---|---|---|---|---|---|---|---|---|---|
| Number of PLLs | 0 | 1 | 2 | 2 | 1 | 2 | 2 | 1 | 1 | 1 |
| Number of Global Buffers | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |

1. The following device packages do not have PLLs in the LP/HX devices: CM36, CM36A, QN84 and VQ100, and the CM81 on the LP4K and LP8K has one PLL.

## Global Routing Resources

The iCE40 device has eight high drive buffers called global buffers (GBUFx). These are connected to eight low-skew global lines, designed primarily for clock distribution but are also useful for other high-fanout signals such as set/reset and enable signals.

***Figure 17-1. High-drive, Low-skew, High-fanout Global Buffer Routing Resources***




1. GBUF7 and its associted PIO are best for direct differential clock inputs.

The input (sources) to the GBUFx can be:

- Global buffer inputs (GBINx, Gx)
- Programmable interconnect[1]
- PLL output[1]
- Programmable input/output block (PIO)[1]
- Strobe Generators (HSSG, LPSG. For iCE40LM devices only)

1. To use a global buffer along with a user interface or PIO, use the SB_GB primitive if it is not inferred automatically.

The associated GBINx/Gx pin represents the best pin to drive a global buffer from an external source.

## Verilog Instantiation

```
SB_GB My_Global_Buffer_i (// required for a user's internally generated FPGA signal that is
heavily loaded and requires global buffering. For example, a user's logic-generated clock.

.USER_SIGNAL_TO_GLOBAL_BUFFER (Users_internal_Clk),

.GLOBAL_BUFFER_OUTPUT ( Global_Buffered_User_Signal) );
```

## VHDL Instantiation

```
component SB_GB
   port (
   USER_SIGNAL_TO_GLOBAL_BUFFER  :     input std_logic;
   GLOBAL_BUFFER_OUTPUT          :     output std_logic);
end component;

My_Global_Buffer_i: SB_GB
   port map (
   USER_SIGNAL_TO_GLOBAL_BUFFER  =>    Users_internal_Clk,
   BUFFER                        =>    Global_Buffered_User_Signal);
```

Refer to the iCE Technology Library for more details on device primitives

If not used in an application, individual global buffers are turned off to save power.

Table 17-2 lists the connections between a specific global buffer and the inputs on a Programmable Logic Block (PLB). Refer to the Architecture section of the iCE40 Family Data Sheet for more information on PLBs. All global buffers optionally connect to all clock inputs. Any four of the eight global buffers can drive logic inputs to a PLB. Even-numbered global buffers optionally drive the reset input to a PLB. Similarly, odd-numbered buffers optionally drive the PLB clock-enabled input.

***Table 17-2. Global Buffer Connections to a Programmable Logic Block***

| Global Buffer | LUT Inputs | Clock | Clock Enable | Reset |
|---|---|---|---|---|
| GBUF0 | Yes, any 4 of 8 GBUF Inputs | ✓ | ✓ | |
| GBUF1 | | ✓ | | ✓ |
| GBUF2 | | ✓ | ✓ | |
| GBUF3 | | ✓ | | ✓ |
| GBUF4 | | ✓ | ✓ | |
| GBUF5 | | ✓ | | ✓ |
| GBUF6 | | ✓ | ✓ | |
| GBUF7 | | ✓ | | ✓ |

Table 17-3 lists the connections between a specific global buffer and the inputs on a Programmable I/O (PIO) pins. Although there is no direct connection between a global buffer and a PIO output, such a connection is possible by first connecting through a PLB LUT4 function. Again, all global buffers optionally drive all clock inputs. However, even-numbered global buffers optionally drive the clock-enable input on a PIO pair.

***Table 17-3. Global Buffer Connections to Programmable I/O Pair***

| Global Buffer | Output Connections | Input Clock | Output Clock | Clock Enable |
|---|---|---|---|---|
| GBUF0 | None (connect through PLB LUT) | ✓ | ✓ | ✓ |
| GBUF1 | | ✓ | ✓ | |
| GBUF2 | | ✓ | ✓ | ✓ |
| GBUF3 | | ✓ | ✓ | |
| GBUF4 | | ✓ | ✓ | ✓ |
| GBUF5 | | ✓ | ✓ | |
| GBUF6 | | ✓ | ✓ | ✓ |
| GBUF7 | | ✓ | ✓ | |

# iCE40 sysCLOCK PLL

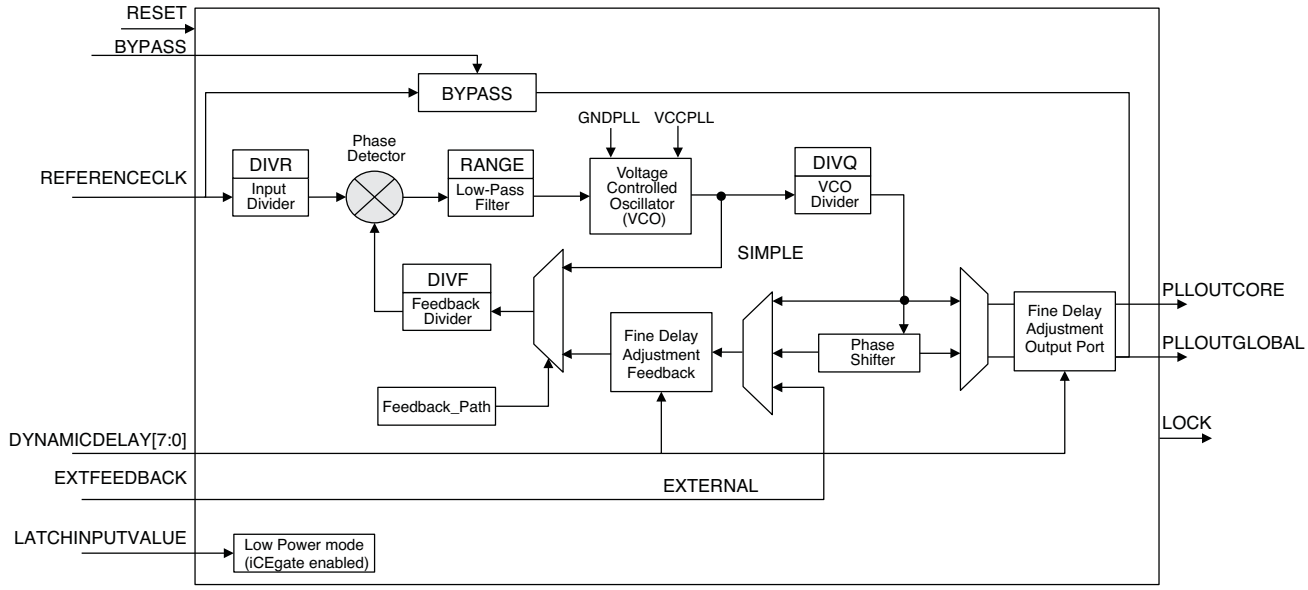
The iCE40 Phase Locked Loop (PLL) provides a variety of user-synthesizable clock frequencies, along with custom phase delays.The PLL in the iCE40 device can be configured and utilized with the help of software macros or the PLL Module Generator. The PLL Module Generator utility helps users to quickly configure the desired settings with the help of a GUI and generate Verilog code which configures the PLL macros. Figure 17-2 shows the iCE40 sysCLOCK PLL block diagram.

## iCE40 sysCLOCK PLL Features

The PLL provides the following functions in iCE40 applications:

- Generates a new output clock frequency
  - Clock multiplication
  - Clock division
- De-skews or phase-aligns an output clock to the input reference clock
  - Faster input set-up time
  - Faster clock-to-output time
- Corrects output clock to have nearly a 50% duty cycle, which is important for Double Data Rate (DDR) applications
- Optionally phase shifts the output clock relative to the input reference clock
  - Optimal data sampling within the available bit period
  - Fixed quadrant phase shifting at 0°, 90°
  - Optional fine delay adjustments of up to 2.5 ns (typical) in 150 ps increments (typical)

*Figure 17-2. iCE40 Phase Locked Loop (sysCLOCK PLL) Block Diagram*




## Signals

Table 17-4 lists the signal names, direction, and function of each connection to the PLL. Some of the signals have an associated attribute or property, as listed in Table 17-4. Table 17-4 lists the attributes or properties associated with the PLL and the allowable settings for each attribute.

*Note: Signals and attribute settings of PLL primitives are for reference only. It is recommend to generate a PLL module with the GUI-based PLL Module Generator as explained in "Generating iCE40 PLL Using PLL Module Generator in iceCube2 Design Software" on page 8.*

***Table 17-4. PLL Signals***

| Signal Name | Direction | Description |
|---|---|---|
| REFERENCECLK | Input | Input reference clock |
| RESET | Input | Reset |
| BYPASS | Input | When FEEDBACK_PATH is set to SIMPLE, the BYPASS control selects which clock signal connects to the PLLOUT output.<br>0 = PLL generated signal<br>1 = REFERENCECLK |
| EXTFEEDBACK | Input | External feedback input to PLL. Enabled when the FEEDBACK_PATH attribute is set to EXTERNAL. |
| DYNAMICDELAY[3:0] | Input | Fine delay adjustment control inputs. Enabled when DELAY_ADJUSTMENT_MODE is set to DYNAMIC. |
| LATCHINPUTVALUE | Input | When enabled, forces the PLL into low-power mode; PLL output is held static at the last input clock value. Set ENABLE ICEGATE_PORTA and PORTB to '1' to enable. |
| PLLOUTGLOBAL | Output | Output from the Phase-Locked Loop (PLL). Drives a global clock network on the FPGA. The port has optimal connections to global clock buffers GBUF4 and GBUF5. |
| PLLOUTCORE | Output | Output clock generated by the PLL, drives regular FPGA routing. The frequency generated on this output is the same as the frequency of the clock signal generated on the PLLOUTLGOBAL port. |
| LOCK | Output | When High, indicates that the PLL output is phase aligned or locked to the input reference clock. |
| RESET | Input | Active low reset. |

*Table 17-5. PLL Attributes and Settings in PLL Macro [1]*

| Parameter Name | Description | Parameter Value | Description |
|---|---|---|---|
| FEEDBACK_PATH | Selects the feedback path to the PLL | SIMPLE | Feedback is internal to the PLL, directly from VCO. |
| | | DELAY | Feedback is internal to the PLL, through the Fine Delay Adjust Block. |
| | | PHASE_AND_DELAY | Feedback is internal to the PLL, through the Phase Shifter and the Fine Delay Adjust Block. |
| | | EXTERNAL | Feedback path is external to the PLL and connects to the EXTFEEDBACK pin. Also uses the Fine Delay Adjust Block. |
| DELAY_ADJUSTMENT_MODE_FEEDBACK | Selects the mode for the Fine Delay Adjust block in the feedback path | FIXED | Delay of the Fine Delay Adjust Block is fixed, the value is specified by the FDA_FEEDBACK parameter setting. |
| | | DYNAMIC | Delay of Fine Delay Adjust Block is determined by the signal value at the DYNAMICDELAY[3:0] pins. |
| FDA_FEEDBACK | Sets a constant value for the Fine Delay Adjust Block in the feedback path | 0, 1,…,15 | The PLLOUTGLOBAL and PLLOUTCORE signals are delay compensated by (n+1)*150 ps, where n = FDA_FEEDBACK only if DELAY_ADJUSTMENT_MODE_FEEDBACK is set to FIXED. |
| DELAY_ADJUSTMENT_MODE_RELATIVE | Selects the mode for the Fine Delay Adjust block | FIXED | Delay of the Fine Delay Adjust Block is fixed, the value is specified by the FDA_RELATIVE parameter setting. |
| | | DYNAMIC | Delay of the Fine Delay Adjust Block is determined by the signal value at the DYNAMICDELAY[7:4] pins. |
| FDA_RELATIVE | Sets a constant value for the Fine Delay Adjust Block | 0, 1,…,15 | The PLLOUTGLOBALA and PLLOUTCOREA signals are additionally delayed by (n+1)*150 ps, where n = FDA_RELATIVE. Used if DELAY_ADJUSTMENT_MODE_RELATIVE is set to FIXED. |
| SHIFTREG_DIV_MODE | Selects shift register configuration | 0,1 | Used when FEEDBACK_PATH is set to PHASE_AND_DELAY.<br>0 = Divide by 4<br>1 = Divide by 7 |

*Table 17-5. PLL Attributes and Settings in PLL Macro (Continued)*[1]

| Parameter Name | Description | Parameter Value | Description |
|---|---|---|---|
| PLLOUT_SELECT | Selects the signal to be output at the PLLOUTCORE and PLLOUTGLOBAL ports | SHIFTREG_0deg | 0° phase shift only if the setting of FEEDBACK_PATH is set to PHASE_AND_DELAY. |
| | | SHIFTREG_90deg | 90° phase shift only if the setting of FEEDBACK_PATH is PHASE_AND_DELAY and SHIFTREG_DIV_MODE = 0. |
| | | GENCLK | The internally generated PLL frequency will be output without any phase shift. |
| | | GENCLK_HALF | The internally generated PLL frequency will be divided by 2 and then output. No phase shift. |
| DIVR | REFERENCECLK divider | 0,1,2,…,15 | These parameters are used to control the output frequency, depending on the FEEDBACK_PATH setting. |
| DIVF | Feedback divider | 0,1,..,63 | |
| DIVQ | VCO divider | 1,2,…,6 | |
| FILTER_RANGE | PLL filter range | 0,1,…,7 | |
| EXTERNAL_DIVIDE_FACTOR | Divide-by factor of a divider in external feedback path | User specified value. Default = 1 | Specified only when there is a user-implemented divider in the external feedback path. |
| ENABLE_ICEGATE | Enables the PLL power-down control | 0 | Power-down control disabled. |
| | | 1 | Power-down controlled by the LATCHINPUTVALUE input. |

1. The attributes are automatically configured through the PLL Module Generator.

## Clock Input Requirements

Proper operation requires the following considerations:

- A stable monotonic (single frequency) reference clock input
- The reference clock input must be within the input clock frequency range, $F_{REF}$, specified in the data sheet
- The reference clock must have a duty cycle that meets the requirement specified in the data sheet
- The jitter on the reference input clock must not exceed the limits specified in the data sheet

## PLL Output Requirements

The PLL output clock, PLLOUT, has the following restrictions:

- The PLLOUT output frequency must be within the limits specified in the data sheet
- The PLLOUT output is not valid or stable until the PLL LOCK output remains high

## Functional Description

The PLL optionally multiplies and/or divides the input reference clock to generate a PLLOUT output clock of another frequency. The output frequency depends on the frequency of the REFERENCLK input clock and the settings for the DIVR, DIVF, DIVQ, RANGE, and FEEDBACK_PATH attribute settings, as indicated in Figure 17-2.

The PLL's phase detector and Voltage Controlled Oscillator (VCO) synthesize a new output clock frequency based on the attribute settings. The VCO is an analog circuit and has independent voltage supply and ground connections labeled VCCPLL and GNDPLL.

**PLLOUT Frequency for All Modes Except FEEDBACK_PATH = SIMPLE**

For all the FEEDBACK_PATH modes, except SIMPLE, the PLLOUT frequency calculated as per the equation below.

$$F_{PLLOUT} = \frac{F_{REFERENCECLK} \times (\mathrm{DIVF} + 1)}{\mathrm{DIVR} + 1}$$

**PLLOUT Frequency for FEEDBACK_PATH = SIMPLE**

In the SIMPLE feedback mode, the PLL feedback signal taps directly from the output of the VCO, before the final divider stage. Consequently, the PLL output frequency has an additional divider step, DIVQ, contributed by the final divider step as shown in equation below. (DIVF, DIVQ and DIVR are binary).

$$F_{PLLOUT} = \frac{F_{REFERENCECLK} \times (\mathrm{DIVF} + 1)}{2^{(DIVQ)} \times (DIVR + 1)}$$

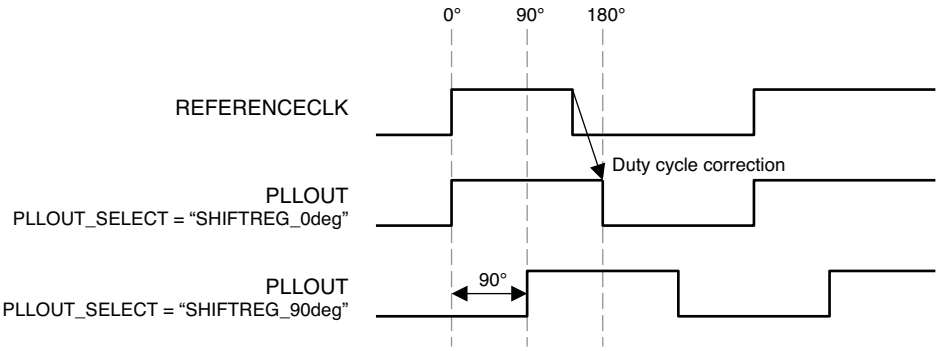
**Fixed Quadrant Phase Shift**

The PLL optional phase feature shifts the PLLOUT output by a specified quadrant or quarter clock cycle as shown in Figure 17-3 and Table 17-6. The quadrant phase shift option is only available when the FEEDBACK_PATH attribute is set to PHASE_AND_DELAY.

***Table 17-6. PLL Phase Shift Options***

| PLLOUT_SELECT | Duty Cycle Correction | Phase Shift | Fraction Clock Cycle |
|---|---|---|---|
| SHIFTREG_0deg | Yes | 0º | None |
| SHIFTREG_90deg | Yes | 90º | Quarter Cycle |

***Figure 17-3. Fixed Quadrant Phase Shift Control***




Unlike the Fine Delay Adjustment, the quadrant phase shifter always shifts by a fixed phase angle. The resulting phase shift, measured in delay, depends on the clock period and the PLLOUT_PHASE phase shift setting, as shown in the equation below.

$$Delay = \frac{\text{Phase Shift}}{360°} \times \text{Clock\_Period}$$

**Fine Delay Adjustment (FDA)**

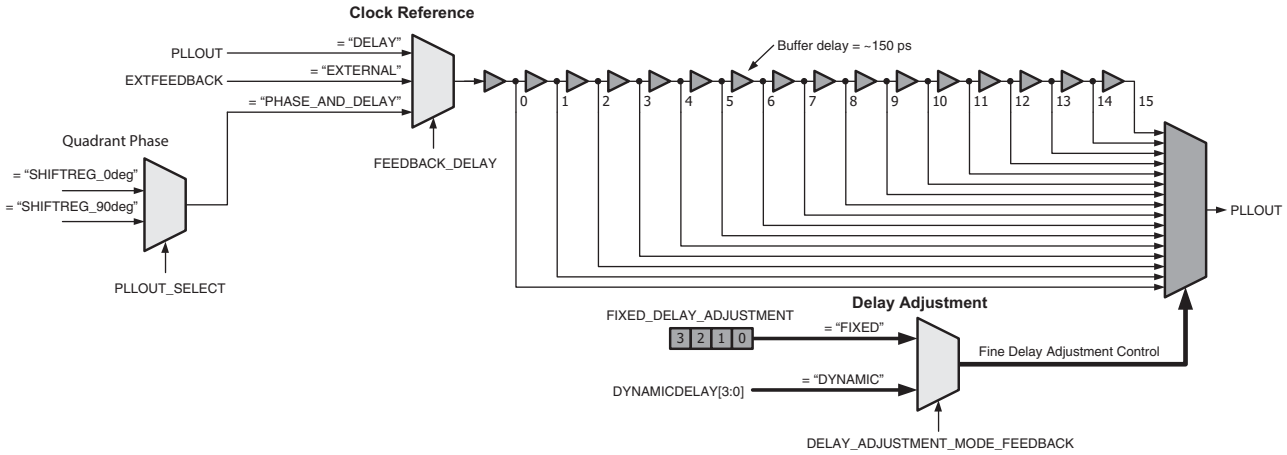
The PLL provides two optional fine delay adjustment blocks that control the delay of the PLLOUT output relative to the input reference clock, to an external feedback signal, or relative to the selected quadrant phase shifted clock. One FDA is placed in the feedback path, while the other FDA provides delay on the output port directly. If a two-port PLL is used, this additional delay is applied only on Port A. Unlike the Feedback FDA, the output port FDA is not dependent on FEEDBACK_PATH, and can be used even if FEEDBACK_PATH = Simple. The PLL Module Generator provides easy selection of the two fine delay adjust blocks. Figure 17-4 shows the typical first fine delay adjust control block.

The delay is adjusted by selecting one or more of the 16 delay taps inside the fine delay adjustment block. Each tap is approximately 150 ps.

Fine Delay Adjustment (nominal) = (n+1) * 150ps; 0 ≤ n ≤ 15, where 'n' is the number of delay taps.

The number of taps can be selected statically (by providing the value within the PLL Module Generator) or dynamically by setting the values in DYNAMICDELAY [7:0]. DYNAMICDELAY [3:0] sets the tap numbers for the feedback path fine delay adjustment block while DYNAMICDELAY [7:0] sets values for the output port FDA. Refer to parameters DELAY_ADJUSTMENT_MODE_FEEDBACK and DELAY_ADJUSTMENT_MODE_RELATIVE in Table 17-2 for more details.

***Figure 17-4. Fine Delay Adjust Control***




**Phase Angle Equivalent**

The fine delay adjustment feature injects an actual delay value, rather than a fixed phase angle like the Fixed Quadrant Phase Shift feature. Use the equation below to convert the fine adjustment delay to a resulting phase angle.

$$\text{Phase Shift} = \frac{\text{Fine\_Delay\_Adustment}}{\text{Clock Period}} \times 360°$$

**Low Power Mode**

The iCE40 sysCLOCK PLL has low operating power by default. The PLL can be dynamically disabled to further reduce power. The low-power mode must first be enabled by setting the ENABLE_ICEGATE attribute to '1'. Once enabled, use the LATCHINPUTVALUE to control the PLL's operation, as shown in Table 17-7. The PLL must reacquire the input clock and LOCK when the LATCHINPUTVALUE returns from '1' to '0', external feedback is used and path goes out into the fabric.

***Table 17-7. PLL LATCHINPUTVALUE Control***

| ENABLE_ICEGATE Attribute | LATCHINPUTVALUE Input | Function |
|---|---|---|
| 0 | Don't Care | PLL is always enabled. |
| 1 | 0 | PLL is enabled and operating. |
| | 1 | PLL is in low-power mode; PLLOUT output holds last clock state. |

# Generating iCE40 PLL Using PLL Module Generator in iceCube2 Design Software

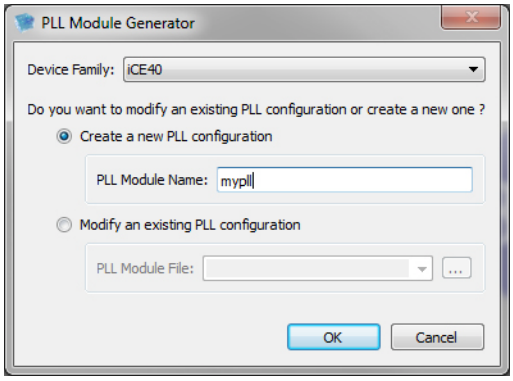
A GUI-based PLL Configuration tool is provided in iceCube2™ design software which configures the iCE40 PLL software macros based on the inputs in the GUI. The resultant HDL code can be used for synthesis.

Figure 17-5 shows the iceCube2 design software. The PLL module generator GUI can be invoked from the Tool menu as shown.

*Figure 17-5. iceCube2 Design Software*

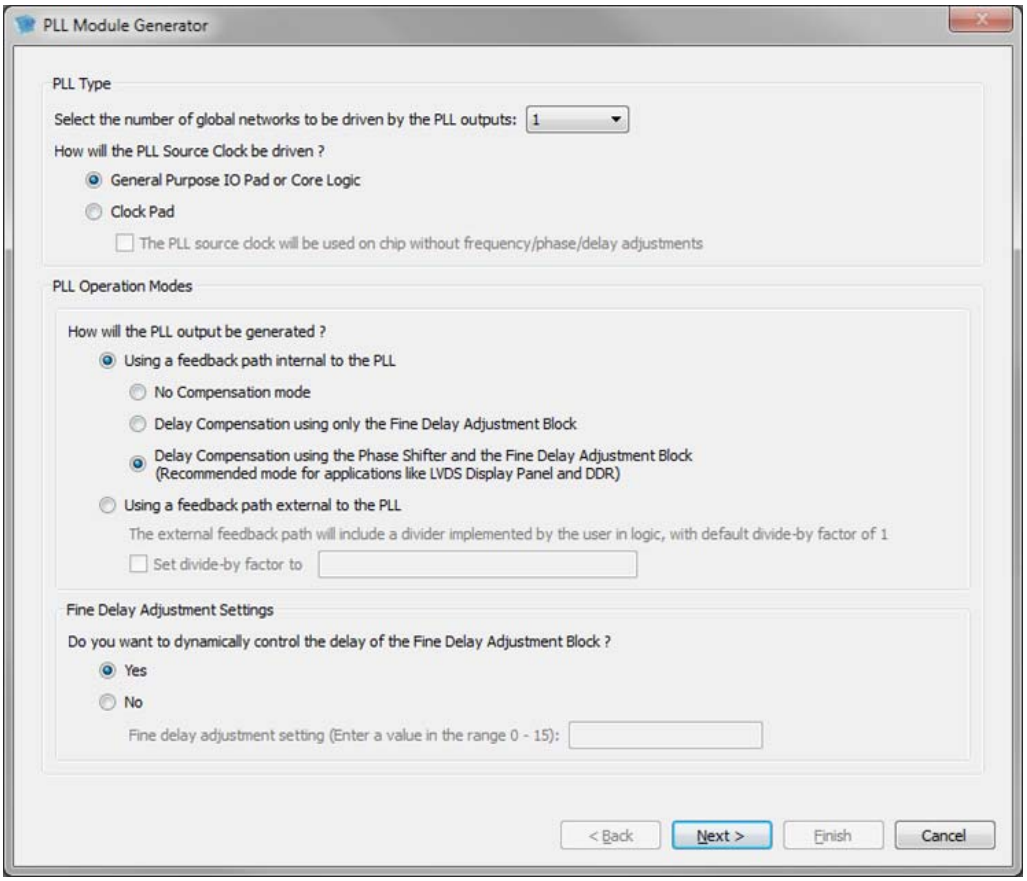

Figure 17-6 shows the PLL configuration GUI. Select the device (iCE40 in this case) and other desired operations.

*Figure 17-6. PLL Module Generator/Modify Existing PLL Configuration*




Click **OK** and the PLL frequency settings window will open up, as shown in Figure 17-7.

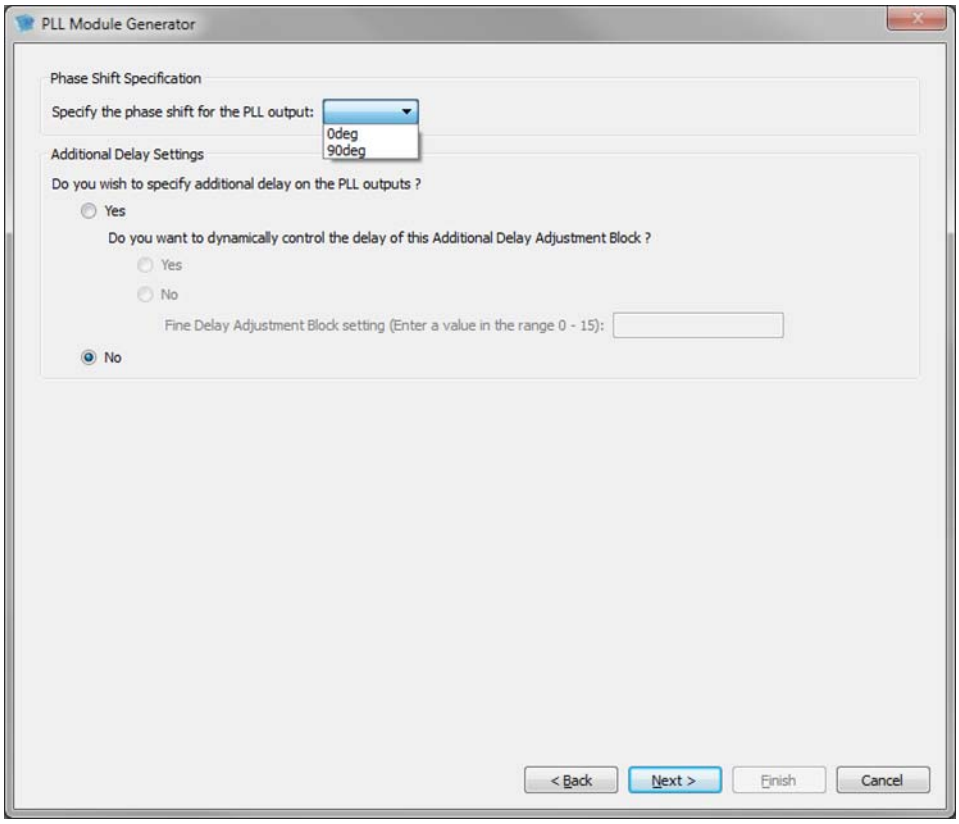
*Figure 17-7. Settings Window 1*




Refer to Table 17-8 for details on user options. Select desired settings which are self-explanatory. Note that some of the options are only activated when other required selections are made. These settings directly modify the PLL signals and attributes of the PLL software macro, as explained in Tables 17-4 and 17-5.

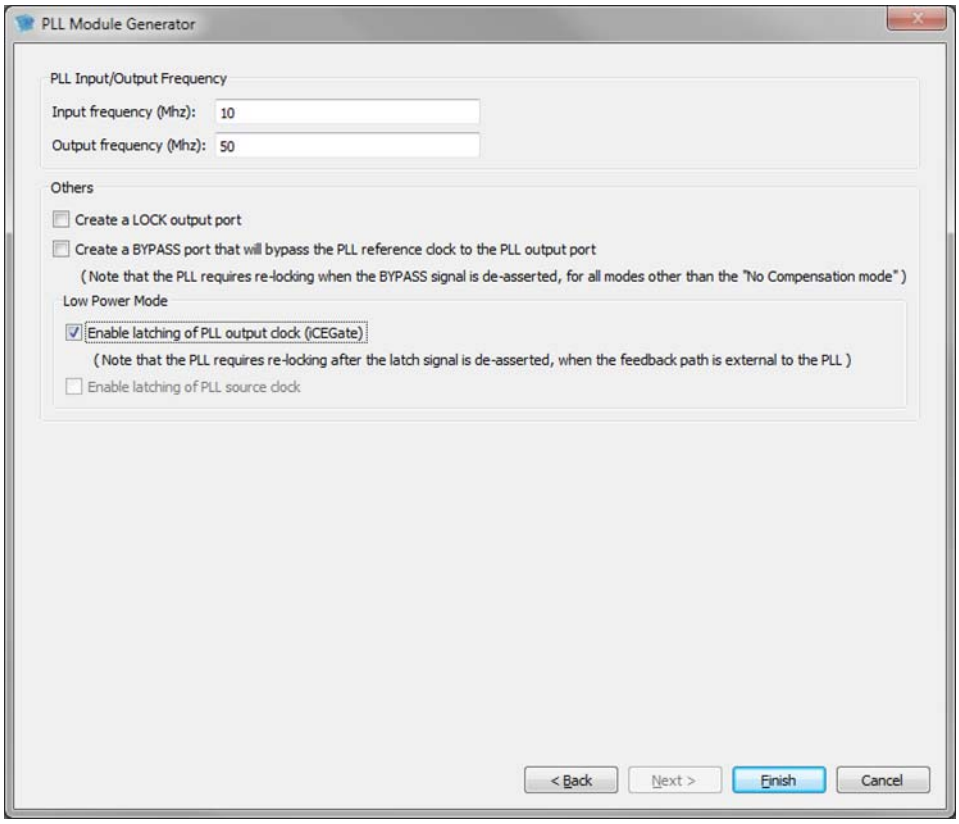
Figure 17-8 shows the next setting window.

*Figure 17-8. Settings Window 2*




Select desired values and click **Next**. The last of the settings windows will open, as shown in Figure 17-9.

*Figure 17-9. Settings Window 3*




After selecting the desired values, the final window the PLL Module Generator opens, as shown in Figure 17-10. This window shows all the values of the attributes and parameters that were discussed in the previous sections and in Tables 17-4 and 17-5. It also shows which PLL Macro type has been selected. The PLL Macro Type used in this example is SB_PLL40_Core.

*Figure 17-10. PLL Configuration – Final Settings*

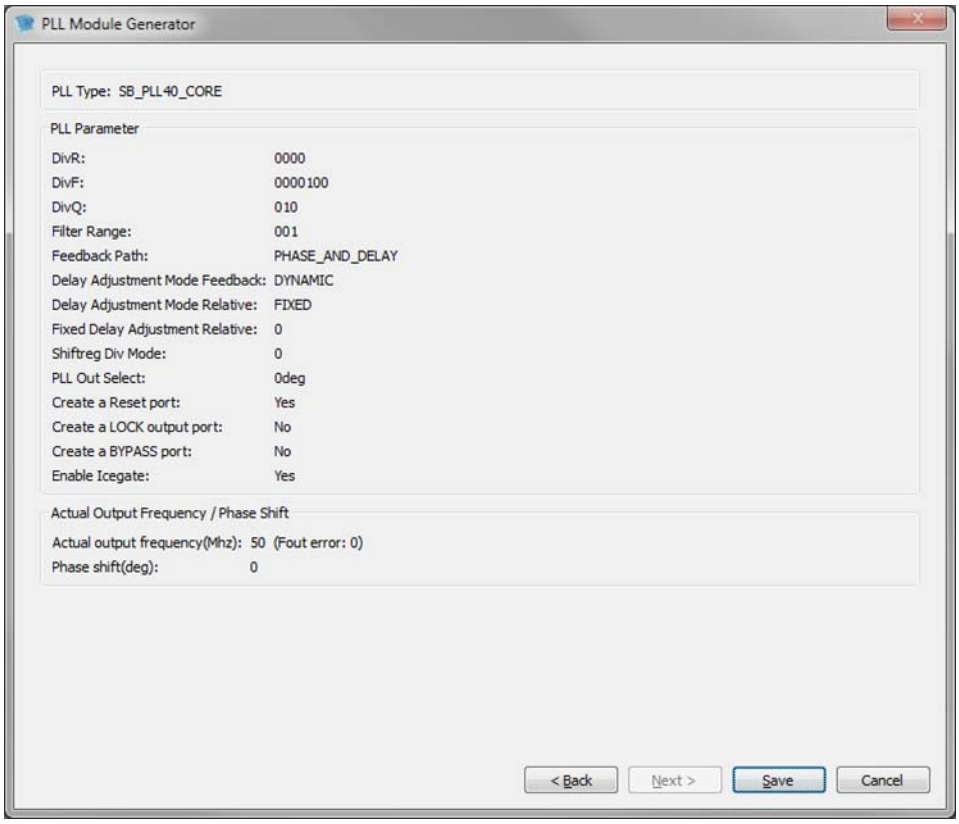



*Table 17-8. PLL Configuration Tool User Parameters*

| User Parameter | Description | Range | Description |
|---|---|---|---|
| **PLL Type** | | | |
| Select the number of global networks to be driven by the PLL outputs | Setting the value to '1' generates a PLL which drives a single global clock network, as well as regular routing. Setting the value to '2' generates a PLL which drives two global clock networks as well as two regular routing resources. | 1 | |
| | | 2 | |
| How will the PLL Source Clock be driven? | | General Purpose I/O Pad or Core Logic | In this scenario, the PLL input (source clock) is driven by a signal from the FPGA fabric. This signal can either be generated on the FPGA core, or it can be an external signal that was brought onto the FPGA using a General Purpose I/O pad. |
| | | Clock Pad | The PLL input clock (source) is driven by a dedicated clock pad located in I/O Bank 2 (bottom bank) or I/O Bank 0 (top bank). If the number of global networks is two, the source clock of the PLL can be used as is (i.e. without any frequency, delay compensation or phase adjustments). It is recommended that if the source clock is required on-chip, this option should not be selected. |

*Table 17-8. PLL Configuration Tool User Parameters (Continued)*

| User Parameter | Description | Range | Description |
|---|---|---|---|
| **PLL Operation Modes** | | | |
| How will the PLL output be generated? | The PLL can be configured to operate in one of multiple modes. An Operation Mode determines the feedback path of the PLL and enables phase alignment of the generated clock with regard to the source clock. | Using a feedback path internal to the PLL | This option is related to the phase delay introduced in the feedback path. The options are:<br>**No Compensation mode** – There is no phase delay in the feedback path.<br>**Delay compensation using only the Fine Delay Adjustment (FDA) Block** – The feedback path traverses through the FDA block, as explained in the section "Fine Delay Adjustment (FDA)" on page 7.<br>**Delay compensation using the phase shifter and the Fine Delay Adjustment (FDA) Block** – For single-port PLL types, the Phase Shifter provides two outputs corresponding to a phase shift of 0º and 90º. For two-port PLL types, the Phase Shifter has two modes: Divide-by-4 mode and Divide-by-7 mode. In Divide-by-4 mode, the output of the B port can be shifted either 0º or 90º with regard to A port outputs. In Divide-by-7 mode, the B port output frequency can be set to have a frequency ratio of 3.5:1 or 7:1 with regard to the port A output frequency. |
| | | Using a feedback path external to the PLL | The feedback path traverses through FPGA routing (external to the PLL) followed by the FDA block. In effect, two delay controls are available – the external path for coarse adjustment and the FDA block for fine delay adjustment. |
| **Fine Delay Adjustment Settings** | | | |
| Do you want to dynamically control the delay of the Fine Delay Adjustment block? | Enabled only when Compensation mode or external Feedback mode is selected. The delay contributed by the FDA block can be fixed or controlled dynamically during FPGA operation. If fixed, it is necessary to provide a number (n) in the range of 0-15 to specify the delay contributed to the feedback path. For further details, see "Fine Delay Adjustment (FDA)" on page 7. | Yes | |
| | | No | Enter a value in the range 0-15. |
| **Phase Shift Specification** | | | |
| Specify the phase shift for the PLL output | Enabled only when delay compensation using the phase shifter is selected. Gives a phase shift of 0º and 90º to the output clock. | 0º | |
| | | 90º | |
| **Target Application**[1] | | | |
| LVDS Display Panel | Two different frequencies can be observed on different ports (A and B). | 3.5:1 | The frequency of port B is 3.5x of Port A. |
| | | 7:1 | The frequency of Port B is 7x of Port A. |
| DDR Application | The signal on Port B can be phase shifted by 90º with respect to signal A. | 0º | |
| | | 90º | |

*Table 17-8. PLL Configuration Tool User Parameters (Continued)*

| User Parameter | Description | Range | Description |
|---|---|---|---|
| **Additional Delay Settings** | | | |
| Do you wish to specify additional delay on the PLL outputs? | In addition to Fine Delay Adjustment in the feedback path, the user can specify additional delay on the PLL output ports. | Yes<br>No | The delay contributed by the delay block can be fixed or controlled dynamically during FPGA operation. If fixed, it is necessary to provide a number (n) in the range 0-15 to specify the delay contributed to the feedback path. The delay for a setting 'n' is calculated as follows : FDA delay = (n+1)*0.15 ps, range of n = 0 to 15.<br>*Note: This additional delay is applied on the output of single-port PLL and port A of two-port PLL Types.* |
| **PLL Input/output Frequency** | | | |
| Input Frequency (MHz) | Specify input frequency. Refer to the iCE40 Family Data Sheet for the input range. | | |
| Output Frequency (MHz) | Specify desired output frequency. Refer the iCE40 Family Data Sheet for the output frequency range. | | |
| **Others** | | | |
| Create a LOCK output port | A lock signal is provided to indicate that the PLL has locked on to the incoming signal. Lock asserts High to indicate that the PLL has achieved frequency lock with a good phase lock. | | |
| Create a BYPASS port that will bypass the PLL reference clock to the PLL output port | A BYPASS signal is provided which both powers-down the PLL core and bypasses it such that the PLL output tracks the input reference frequency. | | |
| Low Power Mode | A control is provided to dynamically put the PLL into a lower power mode through the iCEGate feature. The iCEGate feature latches the PLL output signal and prevents unnecessary toggling. | Enable latching of PLL clock (iCEGate) | Dynamically controls power by enabling the signal LATCHINPUTVALUE. Refer to the section "Low Power Mode" on page 8. |
| | | Enable latching of PLL source clock | Default setting |

1. Enabled when the Number of Global Networks to be Driven by the PLL Outputs option is set to '2'.

## PLL Module Generator Output

The PLL module generator generates two HDL files:

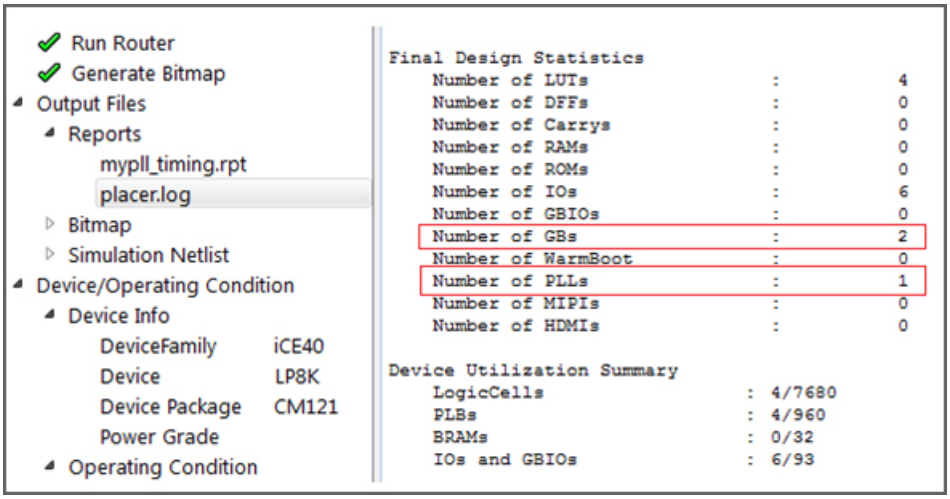
- <module_name>_inst.v
- <module_name>.v

The <module_name>_inst.v is the instantiation template to be used in the custom top level design. The <module_name>.v contains the PLL software macro with the required attributes and signal values, calculated based on the inputs to the GUI.

## iCEcube2 Design Software Report File

The placer.log file (Final Design Statistics section) shows the use of PLLs and GBUFs (global buffers) along with other design elements of the iCE40 device. Note that when GBUF is instantiated in the RTL to connect to a PIO, an extra slice will be utilized for connection.

***Figure 17-11. Report File Showing the Use of PLLs and Global Buffers***




# Hardware Design Considerations

## PLL Placement Rules

- If any instance of PLL is placed in the location of the IO cell, then, an instance of SB_GB_IO cannot be placed in that particular IO cell.
- If an instance of ice40_PLL_CORE or ice40_PLL_2F_CORE is placed, an instance of SB_IO in “output-only” mode can be placed in the associated IO cell location.
- If an instance of ice40_PLL_PAD, ice40_PLL_2F_PAD, ice40_PLL_2_PAD is placed, the associated IO cell cannot be used by any SB_IO or SB_GB_IO.
- If an instance of ice40_PLL_2F_CORE, ice40_PLL_2F_PAD, ice40_PLL_2_PAD is placed, an instance of SB_IO in “output-only” mode can be placed in the right neighboring IO cell.

## Analog Power Supply Filter for PLL

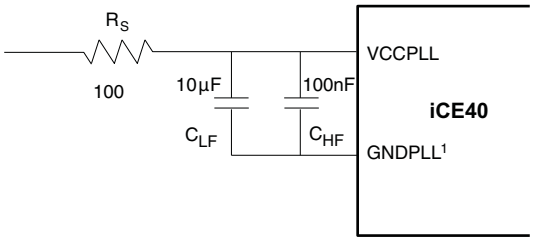
The iCE40 sysCLOCK PLL contains some analog blocks, On some devices, the PLL requires a separate power and ground that is quiet and stable to reduce the output clock jitter of the PLL. On some devices with low pin count (such as iCE40LM in 25WLCSP), the Vccpll is internally connected to Vcc.

On devices with external power and ground for the PLL, an R-C filter as shown in Figure 17-12 is used as a power supply filter on the PLL power and ground pins. The series resistor ($R_S$) limits the voltage drop across the filter. A high frequency non-electrolytic capacitor ($C_{HF}$) is placed in parallel with a lower frequency electrolytic capacitor ($C_{LF}$). $C_{HF}$ is used to attenuate high frequency components while $C_{LF}$ is used for low frequency cut-off.

Board layout around the high frequency capacitor and the path to the pads is critical. The PLL power ($V_{CCPLL}$) path must be a single wire from the FPGA pin to the high frequency capacitor ($C_{HF}$), then to the low frequency capacitor ($C_{LF}$), through the series resistor ($R_S$) and then to board power $V_{CC}$. The distance from the FPGA pin to the high frequency capacitor should be as short as possible. Similarly, the PLL Ground (GNDPLL) path should be from the FPGA pin to the high frequency capacitor ($C_{HF}$) and then to the low frequency capacitor ($C_{LF}$), with the distance from the FPGA pin to the $C_{HF}$ being as short as possible.

The sysCLOCK PLL has the DC ground connection made on the FPGA, so the external PLL ground connection (GNDPLL) must NOT be connected to the board's ground. Figure 17-12 also includes sample values for the components that make up the PLL power supply filter.

***Figure 17-12. Power Supply Filter for VCCPLL and GNDPLL***




1. GNDPLL should not beconnected to the board's ground

## Technical Support Assistance

e-mail: techsupport@latticesemi.com

Internet: www.latticesemi.com

## Revision History

| Date | Version | Change Summary |
|---|---|---|
| September 2012 | 01.0 | Initial release. |
| February 2013 | 01.1 | Updated range of DIVQ divider settings. |
| September 2013 | 01.2 | Added PLL Placement Rules section. |
| | | Updated Technical Support Assistance information. |
| October 2013 | 01.3 | Added iCE40LM information. |

# iCE40 Hardware Checklist

October 2013 Technical Note TN1252

## Introduction

When designing complex hardware using the iCE40™ device (including iCE40LP/HX and iCE40LM families), designers must pay special attention to critical hardware configuration requirements. This technical note steps through these critical hardware requirements related to the iCE40 device. This document does not provide detailed step-by-step instructions but gives a high-level summary checklist to assist in the design process.

The iCE40 ultra-low power, non-volatile devices are available in three versions – LP series for low power applications, HX series for high performance applications, and LM series for ultra-low power for mobile applications.

This technical note assumes that the reader is familiar with the iCE40 device features as described in DS1040, iCE40 LP/HX Family Data Sheet and DS1045, iCE40LM Family Data Sheet and the technical notes included in HB1011, iCE40 LP/HX/LM Family Handbook. The critical hardware areas covered in this technical note include:

- Power supplies as they relate to the supply rails and how to connect them to the PCB and the associated system
- Configuration and how to connect the configuration mode selection
- Device I/O interface and critical signals

## Power Supply

The VCC (core supply voltage) VCCIO_2, SPI_VCC and VPP_2V5 determine the iCE40 device's stable condition. These supplies need to be at a valid and stable level before the device can become operational. Refer to the iCE40 and iCE40LM Family Data Sheet for voltage requirements.

***Table 18-1. Power Supply Description and Voltage Levels***

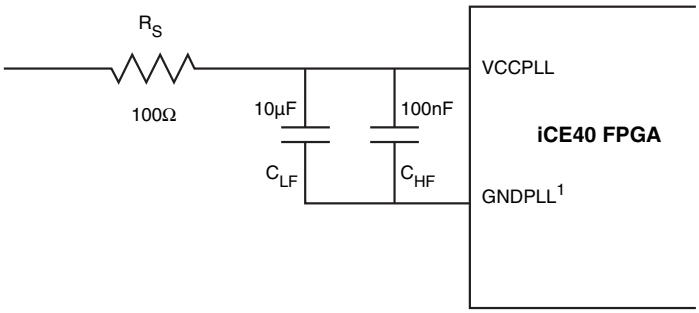
| Supply[3] | Voltage (Nominal Value) | Description |
|---|---|---|
| VCC | 1.20V | Core supply voltage |
| VCCIO_X | 1.5V to 3.3V | Power supply for I/O banks |
| VPP_2V5 | 2.5V | NVCM programming and operating supply voltage |
| VPP_FAST | Leave unconnected | Optional fast NVCM programming supply |
| SPI_VCC | 1.8V to 3.3V | SPI interface supply voltage |
| VCCPLL[1, 2] | 1.2V | Analog voltage supply to Phase Locked Loop (PLL) |

1. VCCPLL must be tied to VCC when PLL is not used.
2. External power supply filter required for VCCPLL and GNDPLL.
3. iCE40LM family devices do not have VPP_2V5 and VPP_FAST supplies.

## Analog Power Supply Filter for PLL

The iCE40 sysCLOCK™ PLL contains analog blocks, so the PLL requires a separate power and ground that is quiet and stable to reduce the output clock jitter of the PLL on device with external VCCPLL supply pins (iCE40LM 25WLCSP connects PLL supply to internal Vcc). The sysCLOCK PLL has the DC ground connection made on the FPGA, so the external PLL ground connection (GNDPLL) must NOT be connected to the board's ground. Figure 18-1 also includes sample values for the components that make up the PLL power supply filter.

*Figure 18-1. Isolating PLL Supplies*




1. Note that GNDPLL should not be connected to the board's ground.

## Configuration Considerations

The iCE40 LP/HX device contains two types of memory, CRAM (Configuration RAM) and NVCM (Non-volatile Configuration Memory). The iCE40LM device contains only the CRAM. CRAM memory contains the active configuration. The NVCM provides on-chip storage of configuration data. It is one-time programmable and is recommended for mass-production.

For more information, refer to TN1248, iCE40 Programming and Configuration.

The configuration and programming of the iCE40 LP/HX/LM device from external memory is using the SPI port, both in Master and Slave modes. In Master SPI mode, the device configures its CRAM from an external SPI Flash connected to it. In Slave mode, the device can be configured or programmed using the Lattice Diamond® Programmer or embedded processor.

On the iCE40LP/HX family devices, the SPI_SS_B determines if the iCE40 CRAM is configured from an external SPI (SPI_SS_B=0) or from the NVCM (SPI_SS_B=1). This pin is sampled after Power-on-Reset (POR) is released or CRESET_B is held low or toggled (High-Low-High).

*Table 18-2. Configuration Pins*

| Pin Name | Function | Direction | External Termination | Notes |
|---|---|---|---|---|
| CRESET_B | Configuration Reset input, active low. | Input | 10 KOhm pull-up to VCCIO_2. | A low on CRESET_B delay's configuration. |
| CDONE | Configuration Done output from iCE40. | Output | Optional 10 KOhm pull-up to VCCIO_2. | |
| SPI_VCC | SPI interface supply voltage. | Supply | | |
| SPI_SI | SPI serial input to the iCE40, in both Master and Slave modes. | Input | | Released to user I/O after configuration. |
| SPI_SO | SPI serial output from the iCE40, in both Master and Slave modes. | Output | | Released to user I/O after configuration. |

*Table 18-2. Configuration Pins (Continued)*

| Pin Name | Function | Direction | External Termination | Notes |
|---|---|---|---|---|
| SPI_SCK | SPI clock | Input/Output | 10 KOhm pull-up to VCCIO_2 recommended. | Direction based on Master or Slave modes. Released to user I/O after configuration. |
| SPI_SS_B | Chip select | Input (Slave mode)/ Output (Master mode) | 10 KOhm pull-up to VCCIO_2 in Master mode and a 10 KOhm pull-down in Slave mode. | Refer to TN1248, iCE40 Programming and Configuration, for more details. |

### SPI Flash Requirement in Master SPI Mode

Users are free to select any industry standard SPI Flash. The SPI Flash must support the 0x0B Fast Read command, using a 24-bit start address with eight dummy bits before the PROM provides first data. Refer to TN1248, iCE40 Programming and Configuration, for additional information.

## LVDS Pin Assignments *(For iCE40LP/HX Devices Only)*

The differential inputs are supported only by Bank 3; however, differential outputs are supported in all banks.

## Checklist

*Table 18-3. iCE40 Hardware Checklist*

| | iCE40 Hardware Checklist Item | OK | N/A |
|---|---|---|---|
| 1 | Power Supply | | |
| 1.1 | Core supply VCC at 1.2V | | |
| 1.2 | I/O power supply VCCIO 0-3 at 1.5V to 3.3V | | |
| 1.3 | SPI_VCC at 1.8V to 3.3V | | |
| 1.4 | VCCPLL pulled to VCC even if PLL not used | | |
| 1.5 | Power supply filter for VCCPLL and GNDPLL | | |
| 1.6 | GNDPLL must NOT be connected to the board | | |
| 2 | Power-on-Reset (POR) inputs | | |
| 2.1 | VCC | | |
| 2.2 | SPI_VCC | | |
| 2.3 | VCCIO_0-3 | | |
| 2.4 | VPP_2V5 | | |
| | VPP_FAST | | |
| 3 | Configuration | | |
| 3.1 | Configuration mode based on SPI_SS_B | | |
| 3.2 | Pull-up on CRESET_B,CDONE pin | | |
| 3.3 | TRST_B is kept low for normal operation | | |
| 4 | I/O pin assignment | | |
| 4.1 | LVDS pin assignment considerations | | |

## Technical Support Assistance

e-mail: techsupport@latticesemi.com

Internet: www.latticesemi.com

## Revision History

| Date | Version | Change Summary |
|---|---|---|
| September 2012 | 01.0 | Initial release. |
| | 01.1 | LVDS Pin Assignments text section – corrected description of differential input and output support. |
| December 2012 | 01.2 | Power Supply Description and Voltage Levels table – corrected VCC nominal voltage. |
| October 2013 | 01.3 | Updated the Configuration Pins table. |
| | | Updated Technical Support Assistance information. |

# Using Differential I/O (LVDS, Sub-LVDS) in iCE40 LP/HX Devices

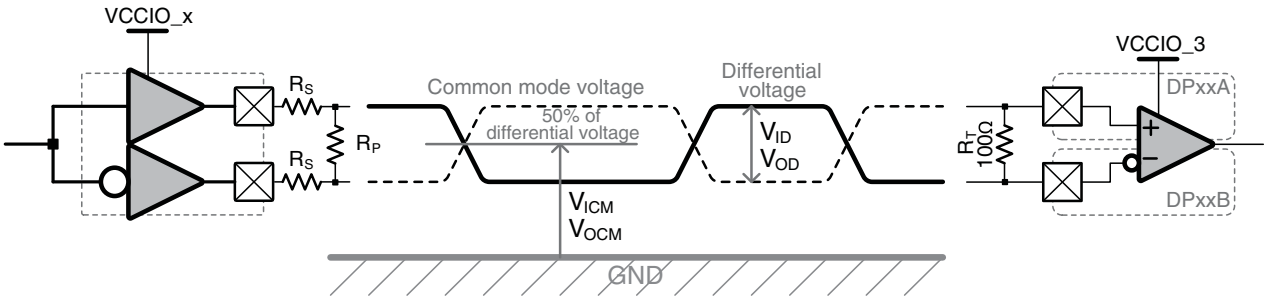
October 2013 Technical Note TN1253

## Introduction

Differential I/O standards are popular in a variety of consumer applications, especially those that require high-speed data transfers such as graphic display drivers and camera interfaces. In these systems, multiple signals are typically combined onto a smaller number of time-division-multiplexed high-speed, differential serial channels.

Differential signals require two Programmable I/O (PIO) pins, working as a pair or a channel, as shown in Figure 19-1. One side of the pair represents the true polarity of the signal while the other side of the pair represents the opposite polarity. The resulting logic value is the difference between the two sides of the signal pair.

***Figure 19-1. Differential Signaling Electrical Parameters***




The key electrical parameters are the common mode voltage and the differential voltage. For iCE40™ applications, the common mode voltage is essentially half the I/O Bank supply voltage. The differential voltage depends on the values of the external compensation resistors, discussed in “LVDS and Sub-LVDS Termination” on page 3.

Differential signaling provides many advantages. In the examples discussed here, all the differential I/O standards have reduced voltage swing, which allows faster switching speeds and potentially higher bandwidth. Reduced voltage swings also mean less dynamic power consumption and reduced electromagnetic interference (EMI).

Differential switching provides **improved noise immunity** and **reduces duty-cycle distortion** caused the differences in rise- and fall-time by the output driver.

The higher potential switching speeds of differential I/O allows data to be multiplexed onto a **reduced number of wires** at a much higher data rate per line. The reduced number of wires reduces system cost and in some cases simplifies the system design. The internal phase-locked loop (PLL) available in iCE40 FPGAs provides convenient on-chip clock multiplication or division to support such applications.

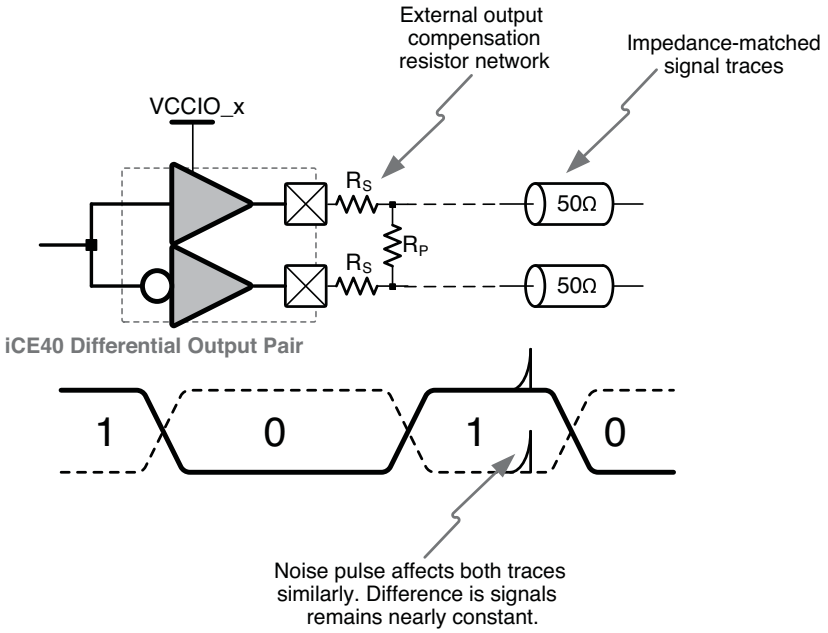
## Differential Outputs

For some differential I/O standards, such as LVDS, the output driver is actually a current source. On iCE40 FPGAs, however, differential outputs are constructed using a pair of single-ended PIO pins as shown in Figure 19-3, and an external resistor network consisting of three resistors. Because differential outputs are built from two single-ended LVCMOS outputs, differential outputs are available in any I/O bank.

The two FPGA outputs must be part of the same I/O tile as indicated in the iCE40 data sheet. The pair choice also depends on the chosen device package as not all I/O tile pairs are bonded out in all packages. Consult the package pinout sections of the iCE40 device data sheets for additional information.

Each differential I/O output pair requires a three-resistor termination network to adjust output characteristics to match those for the specific differential I/O standard. The output characteristics depend on the values of the parallel resistor ($R_P$) and series resistors ($R_S$). These resistors should be surface mounted as close as possible to the FPGA output pins.

***Figure 19-2. Differential Output Pair***



## Differential Inputs

Differential inputs are only supported in I/O Bank 3. The maximum number of differential input pairs per device is shown in Table 19-1 but the actual number available depends on the specific package used.

***Table 19-1. Maximum Differential Input Pairs Available on iCE40 FPGAs***

| | iCE40HX1K | iCE40LP1K | iCE40HX4K | iCE40LP4K | iCE40LP8K | iCE40HX8K |
|---|---|---|---|---|---|---|
| Maximum Differential Input Pairs | 11 | 12 | 12 | 20 | 23 | 26 |

Differential inputs require specific PIO pin pairs as listed in the iCE40 data sheet. Each differential input pair consists of one pin labeled DPxxA and another labeled DPxxB, where “xx” represents the differential pair number. Both pins must be in the same differential pair.

Connect the positive or true polarity side of the differential pair to the DPxxA input and the negative or complementary side of the pair to the DPxxB input. If it is easier to route the differential pair, the input pins can be swapped, which produces an inverted input value. The inverted input value can subsequently be inverted by logic within the FPGA.

An input termination resistor must be connected between the DPxxA and DPxxB pins to generate the differential signal. The resistor’s value must be twice the trace impedance, as described in the following section.

Typically, the resulting signal pair is routed on the printed circuit board (PCB) using controlled impedance and delay matching.

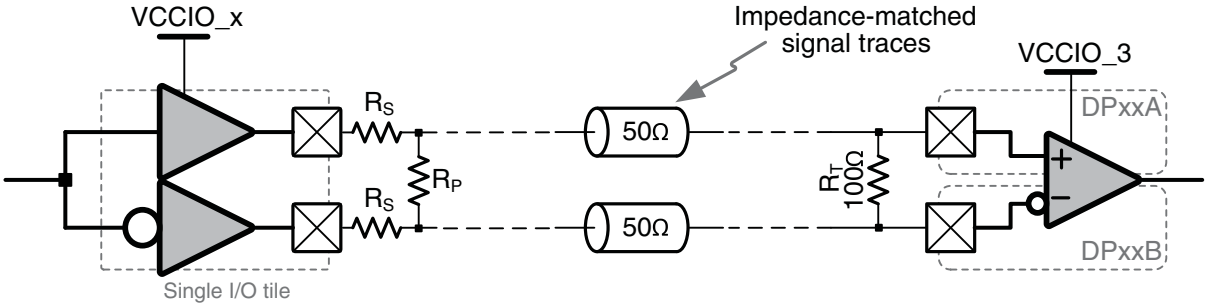
## LVDS and Sub-LVDS Termination

LVDS and Sub-LVDS inputs require external compensation and termination resistors for proper operation, as shown in Figure 19-4. A termination resistor, $R_T$, between the positive and negative inputs at the receiver forms a current loop. The current across this resistor generates the voltage detected by the receiver’s differential input comparator.

Similarly, iCE40 LVDS and Sub-VLDS outputs require an external resistor network, consisting of two series resistors, $R_S$, and a parallel resistor, $R_P$. This resistor network adjusts the FPGA’s output driver to provide the necessary current and voltage characteristic s required by the specification.

The signals are routed with matched trace impedance, $Z_0$, on the printed circuit board, typically with 50 Ω impedance.

***Figure 19-3. iCE40 LVDS or Sub-LVDS Differential I/O Channel***




The resistor values for the compensation network are described below. These equations are also provided in the Differential I/O spreadsheet. The variables are defined and described in Table 19-2.

### Input Termination Resistor ($R_T$)

$$R_T = 2 \times Z_0 \quad (1)$$

### Output Parallel Resistor ($R_P$)

$$R_P = 2 \times \left( \frac{Z_0 \times VCCIO}{VCCIO - (2 \times V_{OD})} \right) \quad (2)$$

### Output Series Resistor ($R_S$)

$$R_S = \left( \frac{Z_0 \times \frac{R_P}{2}}{\frac{R_P}{2} - Z_0} \right) - R_{OUTPUT} \quad (3)$$

***Table 19-2. Compensation Resistor Equation Variables***

| Variable | Description |
|---|---|
| $Z_0$ | Characteristic impedance of the printed circuit board trace; data sheet values assume 50 Ω traces |
| VCCIO | I/O Bank supply voltage, nominal, volts |
| $V_{OD}$ | Differential output voltage swing, nominal, volts |
| $R_{OUTPUT}$ | iCE40 output source resistance, 30 Ω |
| $R_P$ | LVDS/Sub-LVDS output source compensation parallel resistor |
| $R_S$ | LVDS/Sub-LVDS output source compensation series resistor |
| $R_T$ | LVDS/Sub-LVDS input termination resistor |

## Using the Companion iCE40 Differential I/O Calculator Spreadsheet

The iCE40 data sheet recommends specific values for LVDS and Sub-LVDS differential outputs but also assumes that the differential signals are routed with 50 Ω characteristic impedance ($Z_0$). This may not be possible in all applications. Likewise, the system may require some other slightly different I/O standard.

This technical note includes a companion spreadsheet, available for download from the Lattice website and shown in Figure 19-4, to calculate resistor values for non-standard conditions. The values in Figure 19-4 are the default conditions for the LVDS I/O standard. For other standards, simply modify the VCCIO voltage, the differential output voltage, $V_{OD}$, and the characteristic impedance of the printed circuit board traces, $Z_0$.

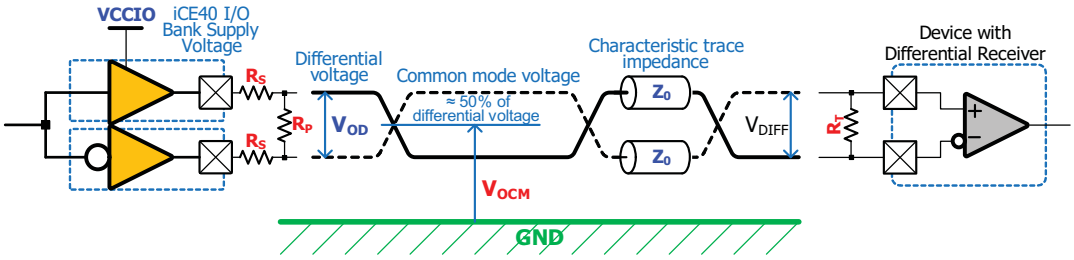
*Figure 19-4. iCE40 Differential I/O Calculator Spreadsheet*

**iCE40™ Differential Output Calculator** (Version 2.3, 28-AUG-2012)

| | I/O Bank Supply | Diff. Output Voltage | Trace Impedance | Common Mode Output Voltage | Source Series Resistor | Source Parallel Resistor | Termination Resistor | Resistor Network Current | Termination Current | Actual Differential Voltage |
|---|---|---|---|---|---|---|---|---|---|---|
| **Symbol** | VCCIO | $V_{OD}$ | $Z_0$ | $V_{OCM}$ | $R_S$ | $R_P$ | $R_T$ | $I_{RES}$ | $I_{RT}$ | $V_{DIFF}$ |
| **Value** | 2.5 | 0.35 | 50 | 1.25 | 150 | 140 | 100 | 6.0 | 3.5 | 0.349 |
| **Units** | volts | volts | ohm | volts | ohm | ohm | ohm | mA | mA | volts |

Companion to Technical Note **TN1253: Using Differential I/O (LVDS, SubLVDS) in iCE40 mobileFPGAs**

***Directions:*** Enter the values for **VCCIO**, $V_{OD}$, and $Z_0$. The resulting resistor values appear under $R_S$, $R_P$, and $R_T$. The $V_{OCM}$ voltage is aslo calculated.




**1.)** The $R_S$ and $R_P$ resistors should be surface mounted as close to the iCE40 mobileFPGA output balls/pins as possible.
**2.)** The termination resistor, $R_T$, should be as close to the recieving device's differential inputs as possible.
**3.)** The actual differential voltage, $V_{DIFF}$, may vary slightly from differential output voltage due to rounding of resistor values.

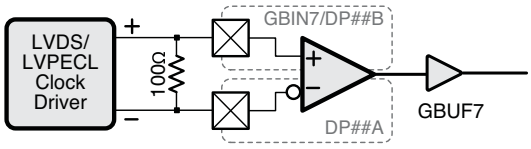
The spreadsheet automatically calculates the common mode output voltage, $V_{OCM}$, which is half of the VCCIO supply voltage. The spreadsheet also automatically calculates the values for the resistor network.

Finally, the spreadsheet also calculates the current draw through the resistor network and for the termination resistor. The spreadsheet rounds the resistor values to the nearest 10 Ω. Consequently, the spreadsheet also calculates the actual differential voltage based on the specified resistor values.

## Differential Clock Input

iCE40 FPGAs have eight global buffers for distributing clocks or other high fanout signals. Global buffer GBUF7, shown in Figure 19-5, is specifically designed to accept a differential clock input on the associated GBIN7/DPxxB, DPxxA differential input pair, which is part of I/O Bank 3. Connect an external 100 Ω termination resistor across the input pair. When VCCIO_3 is 2.5V, this global buffer input accepts either LVDS or LVPECL clock inputs.

*Figure 19-5. LVDS or LVPECL Clock Input*



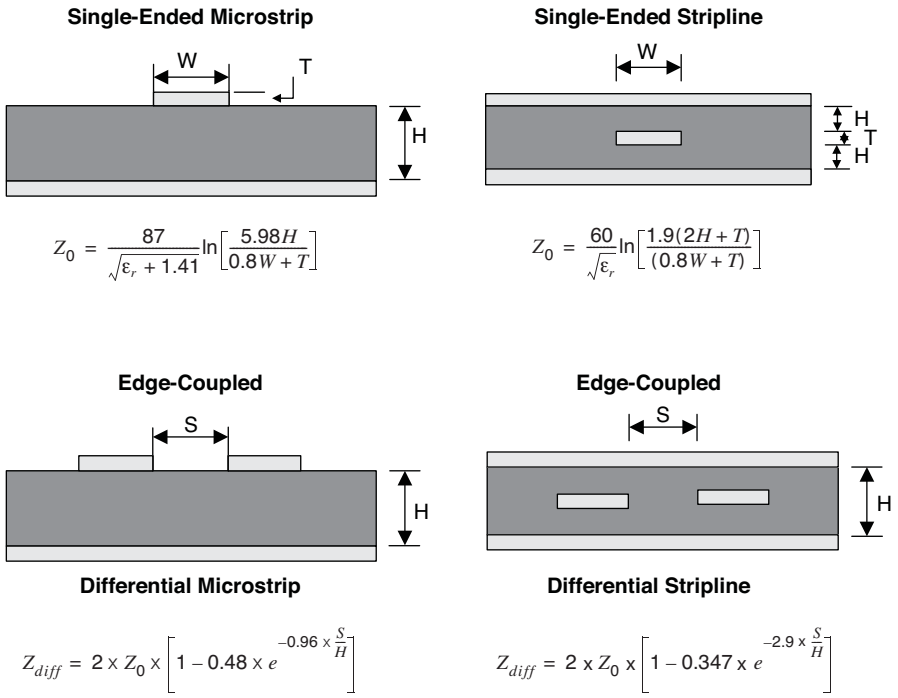

## iCE40 Differential Signaling Board Layout Requirements

Figure 19-6 depicts several transmission line structures commonly used in printed circuit boards (PCBs). Each structure consists of a signal line and a return path with uniform cross-section along its length. The structure's dimensions along with the dielectric material properties determine the characteristic impedance of the transmission line. Figure 19-6 shows examples of edge-coupled microstrips, and edge-coupled or broad-side-coupled striplines.

To maintain constant differential impedance along the length, maintain uniform trace width and spacing, including good symmetry between the two lines.

For differential outputs, place the surface-mounted $R_P$ and $R_S$ resistors as close to the package balls as possible. Similarly, place the 100 Ω termination resistor, $R_T$, as close as possible to the differential input pair.

***Figure 19-6. Controlled Impedance Transmission Lines***




Typical PC board trace impedance is $Z_0$ = 50 Ohms. For a single-ended microstrip, the trace impedance is calculated by using the following equation:

**Single-ended microstrip trace impedance**

$$Z_0 = \frac{87}{\sqrt{\varepsilon_r + 1.41}} \ln\left[\frac{5.98H}{0.8W + T}\right] \quad (4)$$

**Single-ended stripline trace impedance**

$$Z_0 = \frac{60}{\sqrt{\varepsilon_r}} \ln\left[\frac{1.9(2H + T)}{(0.8W + T)}\right] \quad (5)$$

For an LVDS pair, differential impedance can be determined by using the following equations for differential microstrip and differential stripline:

**Differential impedance for differential microstrip**

$$Z_{diff} = 2 \times Z_0 \times \left[1 - 0.48 \times e^{-0.96 \times \frac{S}{H}}\right] \quad (6)$$

**Differential impedance for differential stripline**

$$Z_{diff} = 2 \times Z_0 \times \left[1 - 0.347 \times e^{-2.9 \times \frac{S}{H}}\right] \quad (7)$$

Because the coupling of two traces can lower the effective impedance, use 60 Ω design rules to achieve a differential impedance of approximately 100 ohms.

### Layout Recommendations to Minimize Reflection

Skew delay is introduced if the trace lengths between the signals in the differential pair are not similar. With differing trace lengths, the signals on each side of the differential pair arrive at slightly different times and reflect off the receiver termination, creating a common-mode noise source on the transmission line.

Common-mode noise degrades the receiver's eye diagram, reduces signal integrity, and creates crosstalk between neighboring signals on the board. To minimize reflections due to unmatched trace lengths, consider the following guidelines:

- Match the length of each signal within the differential signal pair to within 20 mils.
- Minimize turns and vias or feed-throughs. Route differential pairs as straight as possible from point-to-point. Do not use 90-degree turns when routing differential pairs. Instead, use 45-degree bevels or rounded curves.
- Minimize vias on or near differential trace lines as these may create additional impedance discontinuities that will increase reflections at the receiver. If vias are required, place them as close to the receiver as possible.
- Use controlled impedance PCB traces. That is, control trace spacing, width, and thickness using stripline or microstrip layout techniques.

## EMI and Noise Cancellation

Differential signaling offers tremendous advantages over single-ended signaling because it is less susceptible to noise. In differential signaling, two current carrying conductors are routed together, with the current in one conductor always equal in magnitude but opposite in direction to the other conductor. The result is that both electromagnetic fields cancel each other.

Common-mode noise rejection is another advantage of differential signaling. The receiver ignores any noise that couples equally on both sides of the differential signal, as shown in Figure 19-8.

***Figure 19-7. Single-ended Input Retains Signal Noise***

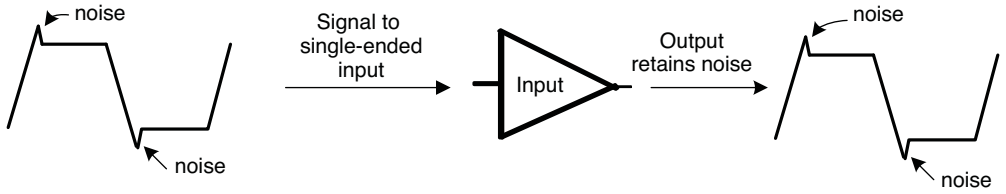
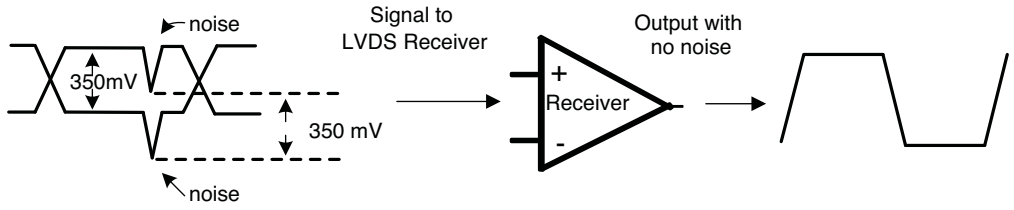



***Figure 19-8. Differential Input Eliminates Common-Mode Noise***




### Reducing EMI Noise

Any skew between differential signals can generate EMI noise on the board. The following are some guidelines to reduce EMI emission onboard:

- Match edge rates and signal skew between differential signals as closely as possible. Different signal rise and fall times and skew between signal pairs create common-mode noise, which generates EMI noise.
- Maintain spacing between differential signal pairs that is less than twice the PCB trace width.

## Defining Differential I/O

Single-ended I/O connections are automatically inferred from the top-level VHDL or Verilog circuit description during logic synthesis. However, differential I/O connections must be specifically instantiated using design primitives from the Lattice technology library for iCE40 FPGAs. Table 19-3 lists the specific I/O primitive required by differential I/O type. Documentation on the design primitives is located under the iCEcube2™ installation directory:

C:\SbtTools\doc\SBT_ICE_Technology_Library.pdf

***Table 19-3. Differential I/O Types and Required SiliconBlue I/O Primitive***

| Differential I/O Type | iCE40 Design Primitive | PIN_TYPE | IO_STANDARD |
|---|---|---|---|
| Differential clock input | SB_GB_IO | See "PIN_TYPE Parameter" on page 9. | SB_LVDS_INPUT |
| Differential input | SB_IO | | SB_LVDS_INPUT |
| Differential output | SB_IO | | SB_LVCMOS |

The SB_IO and SB_GB_IO primitives also require specific parameter settings. Differential inputs always require that IO_STANDARD be set to SB_LVDS_INPUT. Differential outputs typically require that the IO_STANDARD parameter be set to SB_LVCMOS.

### SB_IO Primitive

Table 19-4 lists the signal ports for the SB_IO primitive, which describes one of the Programmable I/O (PIO) pins on an iCE40 FPGA. The table also shows the signal direction for each port, relative to the PIO pin (the SB_IO primitive). These same signals also appear on the SB_GB_IO primitive, which describes a global buffer input.

***Table 19-4. Port Names, Signal Direction, and Description for SB_IO (SB_GB_IO) Primitive***

| Port Name | Direction | Description |
|---|---|---|
| PACKAGE_PIN | I/O | Connection to top-level input, output, or bidirectional signal port. |
| LATCH_INPUT_VALUE | Input | iCEgate latch input. When High, hold the last pad value. Used for power reduction in some PIN_TYPE modes. There is one control input per I/O Bank.<br>0 = Input data flows freely<br>1 = Last data value on pad held constant to save power |
| CLOCK_ENABLE | Input | Clock enable input, shared connection to all flip-flops within the SB_IO primitive. If this port is left unconnected, automatically tied High.<br>0 = Flip-flops hold their current value<br>1 = Flip-flops accept new data on the active clock edge |
| INPUT_CLOCK | Input | Clock for all input flip-flops. If this port is left unconnected, it is automatically tied Low. |
| OUTPUT_CLOCK | Input | Clock for all output flip-flops. If this port is left unconnected, it is automatically tied Low. |
| OUTPUT_ENABLE | Input | Enables the output buffer.<br>0 = Output disabled, pad is high-impedance (Hi-Z)<br>1 = Output enabled, actively driving |
| D_OUT_0 | Input | Data output. For DDR output modes, this is the value clocked out on the rising edge of the OUTPUT_CLOCK. |
| D_OUT_1 | Input | Data output used in DDR output modes. This is the value clocked out on the falling edge of the OUTPUT_CLOCK. |
| D_IN_0 | Output | Data input. For DDR input modes, this is the value clocked into the device on the rising edge of the INPUT_CLOCK. |
| D_IN_1 | Output | For DDR input modes, this is the value clocked into the device on the falling edge of the INPUT_CLOCK. |

## SB_GB_IO Primitive

Global buffer inputs provide a direct connection from a PIO pin to an associated global buffer. This connection can be instantiated using an SB_GB_IO primitive. An SB_GB_IO primitive has all the ports for an SB_IO primitive, shown in Table 19-4, plus the additional connection shown in Table 19-5. Global Buffer Input 7 (GBIN7) is the only one that supports differential clock inputs. See "Differential Clock Input" on page 5 for more information.

***Table 19-5. Additional Port Names on SB_GB_IO Primitive***

| Port Name | Direction | Description |
|---|---|---|
| GLOBAL_BUFFER_OUTPUT | Output | Output from associated Global Buffer. This output is controlled by the iCE-gate latch if a control signal is connected to the iCEgate latch input (LATCH_INPUT_VALUE). |

## PIN_TYPE Parameter

The PIN_TYPE parameter defines the structure and the functionality of any instantiated SB_IO primitive. PIN_TYPE is a six-bit binary value. The upper four bits, PIN_TYPE[5:2], define the output structure while the lower two bits, PIN_TYPE[1:0] define the input structure. Both fields are required, but operate independently.

Global buffer inputs, defined using the SB_GB_IO primitive, are also full-featured PIO pins. However, if only the GLOBAL_BUFFER_OUTPUT is connected on the SB_GB_IO primitive, then the PIN_TYPE parameter has no real effect.

### Output Field (PIN_TYPE[5:2])

Table 19-6 shows the various PIN_TYPE definitions for the output side an SB_IO or SB_GB_IO primitive.

***Table 19-6. Output Structures and PIN_TYPE Values***

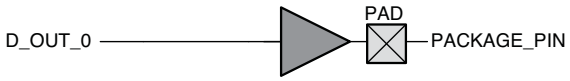
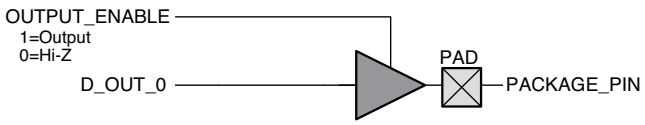
| Style | Mnemonic (Diagram) | PIN_TYPE[5:2] | | | |
|---|---|---|---|---|---|
| None (output disabled) | PIN_NO_OUTPUT   | 0 | 0 | 0 | 0 |
| Non-registered Output | PIN_OUTPUT   | 0 | 1 | 1 | 0 |
| | PIN_OUTPUT_TRISTATE   | 1 | 0 | 1 | 0 |

*Table 19-6. Output Structures and PIN_TYPE Values (Continued)*

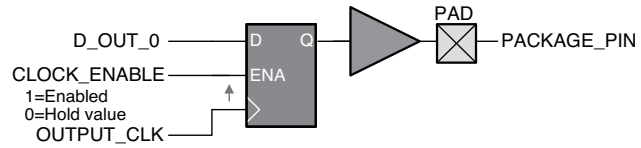
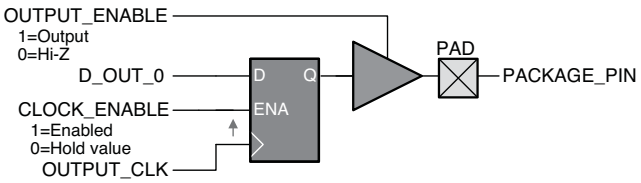
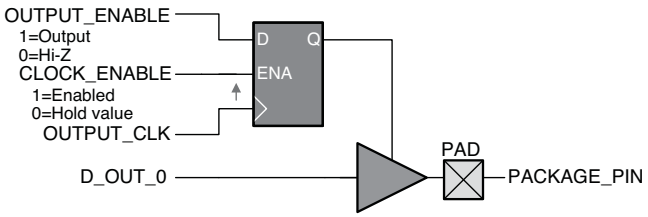
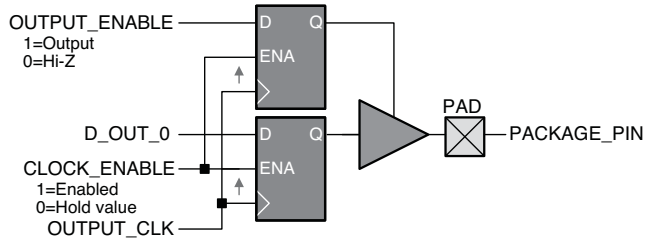
| Style | Mnemonic (Diagram) | PIN_TYPE[5:2] | | | |
|---|---|---|---|---|---|
| Registered Outputs | PIN_OUTPUT_REGISTERED   | 0 | 1 | 0 | 1 |
| | PIN_OUTPUT_REGISTERED_ENABLE   | 1 | 0 | 0 | 1 |
| | PIN_OUTPUT_ENABLE_REGISTERED   | 1 | 1 | 1 | 0 |
| | PIN_OUTPUT_REGISTERED_ENABLE_REGISTERED   | 1 | 1 | 0 | 1 |

*Table 19-6. Output Structures and PIN_TYPE Values (Continued)*

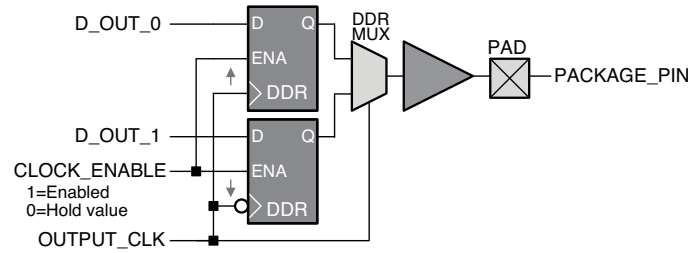
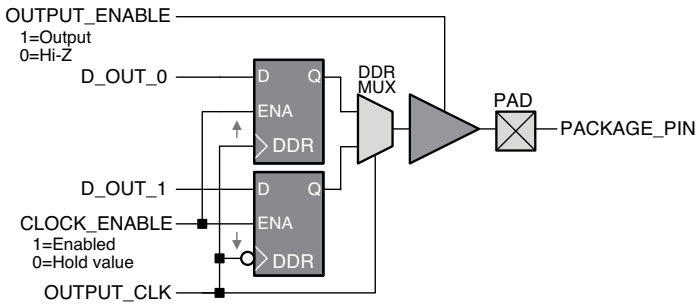
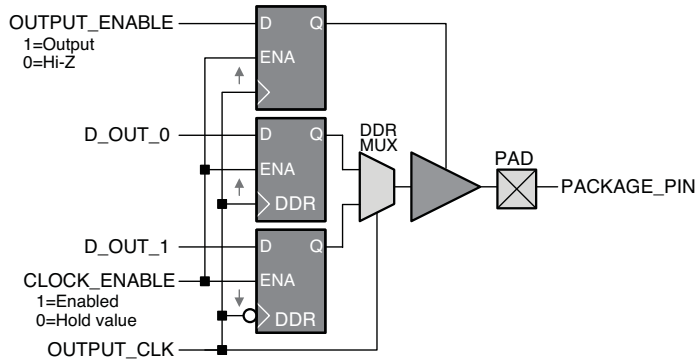
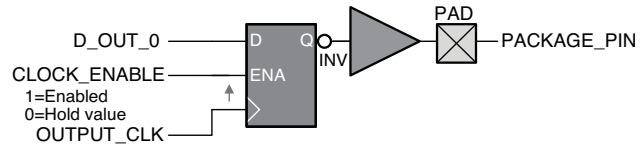
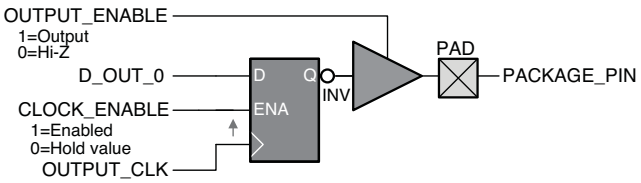
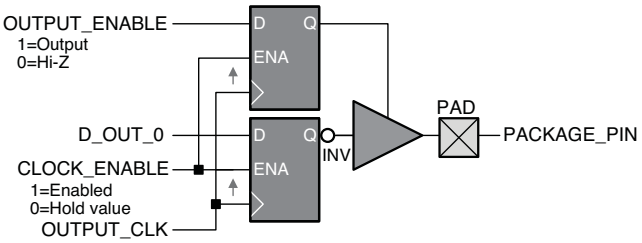
| Style | Mnemonic (Diagram) | PIN_TYPE[5:2] | | | |
|---|---|---|---|---|---|
| Double Data Rage (DDR) Output | PIN_OUTPUT_DDR   | 0 | 1 | 0 | 0 |
| | PIN_OUTPUT_DDR_ENABLE   | 1 | 0 | 0 | 0 |
| | PIN_OUTPUT_DDR_ENABLE_REGISTERED   | 1 | 1 | 0 | 0 |

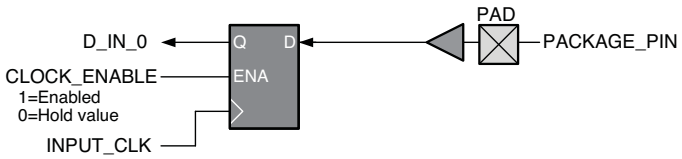
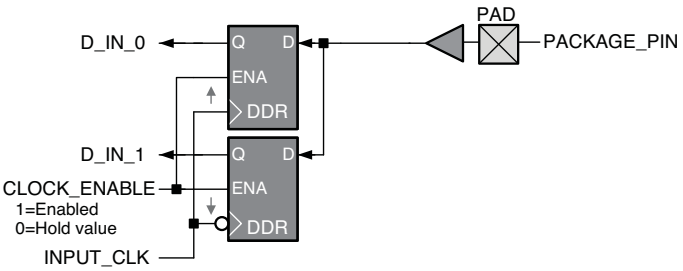
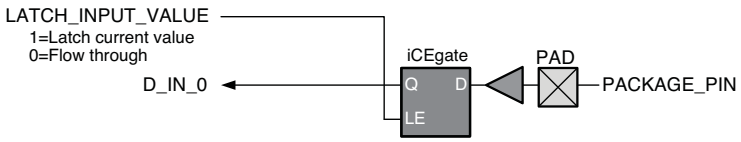
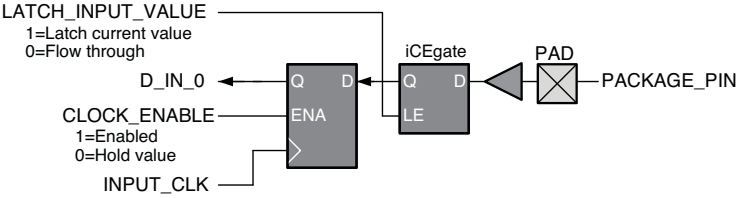
*Table 19-6. Output Structures and PIN_TYPE Values (Continued)*

| Style | Mnemonic (Diagram) | PIN_TYPE[5:2] | | | |
|---|---|---|---|---|---|
| Registered Output, Inverted | PIN_OUTPUT_REGISTERED_INVERTED   | 0 | 1 | 1 | 1 |
| | PIN_OUTPUT_REGISTERED_ENABLE_INVERTED   | 1 | 0 | 1 | 1 |
| | PIN_OUTPUT_REGISTERED_ENABLE_REGISTERED_INVERTED   | 1 | 1 | 1 | 1 |

**Input Field (PIN_TYPE[1:0])**

Table 19-7 shows the various PIN_TYPE definitions for the input side of an SB_IO or SB_GB_IO primitive.

***Table 19-7. Input Structures and PIN_TYPE Values***

| Style | Mnemonic (Diagram) | PIN_TYPE[1:0] | |
|---|---|---|---|
| Direct | PIN_INPUT   | 0 | 1 |
| Registered | PIN_INPUT_REGISTERED   | 0 | 0 |
| Double Data Rage (DDR) Output | PIN_INPUT_DDR   | 0 | 0 |
| iCEgate Low-Power Latch | PIN_INPUT_LATCH   | 1 | 1 |
| | PIN_INPUT_REGISTERED_LATCH   | 1 | 0 |

## IO_STANDARD Parameter

Differential inputs or outputs require specific settings for the IO_STANDARD parameter, as summarized in Table 19-8. Regardless of actual electrical requirements (LVDS, Sub-LVDS), the IO_STANDARD parameter on differential inputs must be set to SB_LVDS_INPUT. Again, differential inputs are only available in I/O bank 3. For differential outputs, set IO_STANDARD to SB_LVCMOS. Differential outputs are available in I/O banks 0, 1, 2, or 3.

***Table 19-8. IO_STANDARD Settings for Differential I/O***

| | Differential Inputs (I/O Bank 3 Only) | Differential Outputs (I/O Banks 0, 1, 2, 3) |
|---|---|---|
| IO_STANDARD Setting | SB_LVDS_INPUT | SB_LVCMOS |

### NEG_TRIGGER Parameter

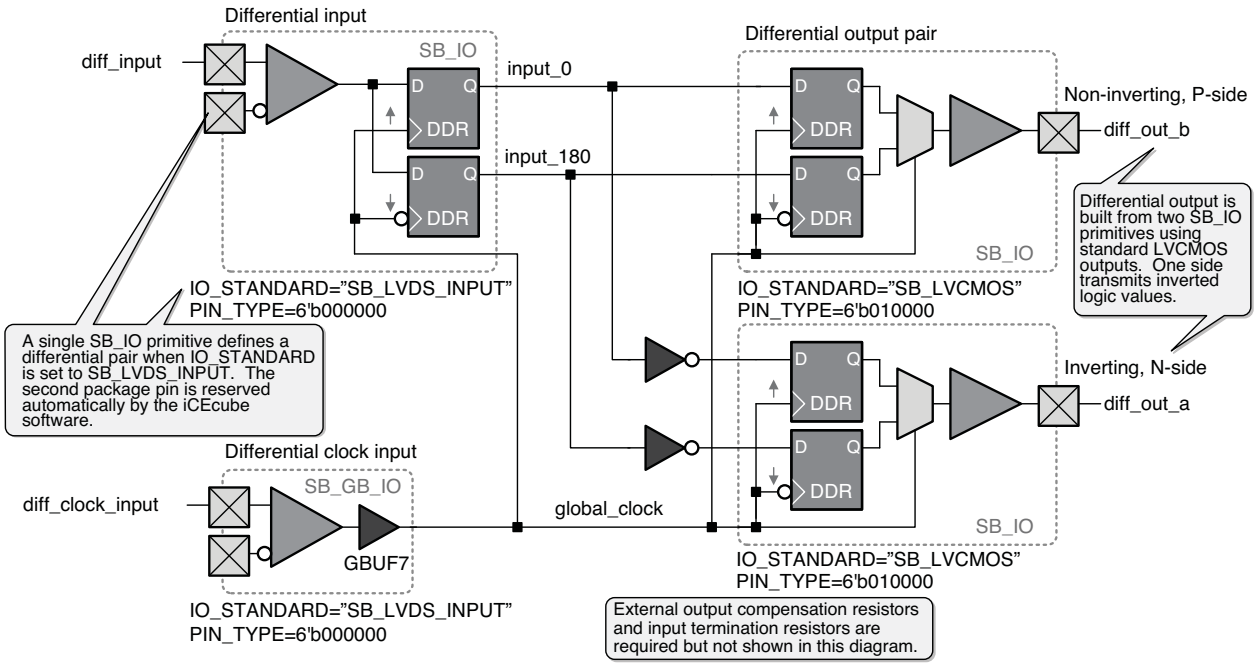
The optional NEG_TRIGGER parameter, when set to '1', inverts the clock polarity within the PIO pin.

## HDL Implementation Example

Most applications that use differential I/O do so because they require very high data bandwidth. Consequently, most of these applications use also Double Data Rate (DDR) flip-flops to double the effective data rate at the PIO pin.

The design example shown in Figure 19-9 uses the DDR flip-flops embedded in every PIO pin.

***Figure 19-9. Differential I/O Design Example***




### VHDL Component Declaration

For VHDL implementations, the SB_IO and SB_GB_IO primitives must be declared as components before they are first used. Verilog does not require this declaration.

### Differential Clock Input

The following Verilog code snippets demonstrate how to instantiate a differential clock input using an SB_GB_IO primitive. The differential clock input is always connected to Global Buffer GBIN7 and always in I/O Bank. In this example, shown in Figure 19-9, the PIO pin only connects an external differential clock signal to the FPGA's GBUF7 global buffer. The only ports connected on the primitive are the PACKAGE_PIN and the GLOBAL_BUFFER_OUTPUT ports. Consequently, the PIN_TYPE setting for this SB_GB_IO primitive does not actually matter.

Set the IO_STANDARD parameter to "SB_LVDS_INPUT". Doing so also causes the iCEcube2 software to reserve a second pin for the other side of the differential input pair.

An external 100 Ω termination resistor must be connected between the two inputs. The resistor must be physically placed as close as possible to the two package balls.

For VHDL implementations, the SB_GB_IO component must be declared.

The PIN_TYPE for this primitive may be different, depending on whether only the global buffer input is used, or if the data input paths are used, or both. The example shown here is for a dedicated differential clock input.

**Verilog**

```
// Differential clock input example: Verilog
// IMPORTANT: The PIN_TYPE is different for a regular LVDS input used as a clock.
//       This example is specifically for the differential clock input.
defparam differential_clock_input.PIN_TYPE = 6'b000000 ; // {NO_OUTPUT, PIN_INPUT_REGISTERED}
defparam differential_clock_input.IO_STANDARD = "SB_LVDS_INPUT" ;
SB_GB_IO differential_clock_input (
  .PACKAGE_PIN(diff_clock_input),
  .LATCH_INPUT_VALUE ( ),
  .CLOCK_ENABLE ( ),
  .INPUT_CLK ( ),
  .OUTPUT_CLK ( ),
  .OUTPUT_ENABLE ( ),
  .D_OUT_0 ( ),
  .D_OUT_1 ( ),
  .D_IN_0 ( ),
  .D_IN_1 ( ),
  .GLOBAL_BUFFER_OUTPUT(global_clock) // Global buffer output
);
```

**VHDL**

Under development.

## Differential Input

The following Verilog and VHDL code snippets demonstrate how to instantiate a differential input using an SB_ IO primitive. Differential inputs are always implemented in I/O Bank 3. In this example, shown in Figure 19-9, the differential input also connects to Double Data Rate (DDR) input flip-flops. There is no output connected. Consequently, set the PIN_TYPE parameter to the binary value “000000”, which defines no output (see Table 19-6) and DDR input (see Table 19-7).

Set the IO_STANDARD parameter to “SB_LVDS_INPUT”. Doing so also causes the iCEcube2 software to reserve a second pin for the other side of the differential input pair.

An external 100 Ω termination resistor must be connected between the two inputs. The resistor must be physically placed as close as possible to the two package balls.

For VHDL implementations, the SB_ IO component must be declared.

**Verilog**

```
// Differential input, DDR data
defparam differential_input.PIN_TYPE = 6'b000000 ; // {NO_OUTPUT, PIN_INPUT_DDR}
defparam differential_input.IO_STANDARD = "SB_LVDS_INPUT" ;
SB_IO differential_input (
  .PACKAGE_PIN(diff_input),
  .LATCH_INPUT_VALUE ( ),
  .CLOCK_ENABLE ( ),
  .INPUT_CLK (global_clock),
  .OUTPUT_CLK ( ),
  .OUTPUT_ENABLE ( ),
  .D_OUT_0 ( ),
  .D_OUT_1 ( ),
  .D_IN_0 (input_0),
  .D_IN_1 (input_180)
);
```

**VHDL**
Under development.

## Differential Output Pair

The following Verilog and VHDL code snippets demonstrate how to instantiate a differential output pair using an SB_ IO primitive. Differential outputs are specified as two separate single-ended outputs. One output provides the non-inverted or P-side of the pair while the other output provides the inverted or N-side of the pair. In this example, shown in Figure 19-9, the differential output also connects to Double Data Rate (DDR) input flip-flops for higher output performance. There is no input connected. Consequently, set the PIN_TYPE parameter to the binary value “010000”, which defines DDR output (see Table ) and a benign input (see Table 19-7).

Differential outputs can be placed in any I/O bank, although the pair must be part of an I/O tile, as shown in the iCE40 or iCE40 data sheet. Because of better slew rate control, place lower-speed differential outputs in I/O Banks 0, 1, or 2. Differential I/Os in I/O Bank 3 have the best performance, but also have faster, noisier switching edges. Set the IO_STANDARD parameter to “SB_LVCMOS”.

An external compensation resistor network must be connected between the two inputs. The resistors must be physically placed as close as possible to the two package balls.

For VHDL implementations, the SB_ IO component must be declared.

**Verilog**

```
// Differential output pair, DDR data
// Non-inverting, P-side of pair
defparam   differential_output_b.PIN_TYPE   =   6'b010000   ;   //   {PIN_OUTPUT_DDR,
PIN_INPUT_REGISTER }
defparam differential_output_b.IO_STANDARD = "SB_LVCMOS" ;
SB_IO differential_output_b (
  .PACKAGE_PIN(diff_output_b),
  .LATCH_INPUT_VALUE ( ),
  .CLOCK_ENABLE ( ),
  .INPUT_CLK ( ),
  .OUTPUT_CLK (global_clock),
  .OUTPUT_ENABLE ( ),
  .D_OUT_0 (input_0),   // Non-inverted
  .D_OUT_1 (input_180),  // Non-inverted
  .D_IN_0 ( ),
  .D_IN_1 ( )
);

// Inverting, N-side of pair
defparam   differential_output_a.PIN_TYPE   =   6'b010000   ;   //   {PIN_OUTPUT_DDR,
PIN_INPUT_REGISTER }
defparam differential_output_a.IO_STANDARD = "SB_LVCMOS" ;
SB_IO differential_output_a (
  .PACKAGE_PIN(diff_output_a),
  .LATCH_INPUT_VALUE ( ),
  .CLOCK_ENABLE ( ),
  .INPUT_CLK ( ),
  .OUTPUT_CLK (global_clock),
  .OUTPUT_ENABLE ( ),
  .D_OUT_0 (~input_0),  // Inverted
  .D_OUT_1 (~input_180), // Inverted
  .D_IN_0 ( ),
  .D_IN_1 ( )
);
```

**VHDL**
Under development.

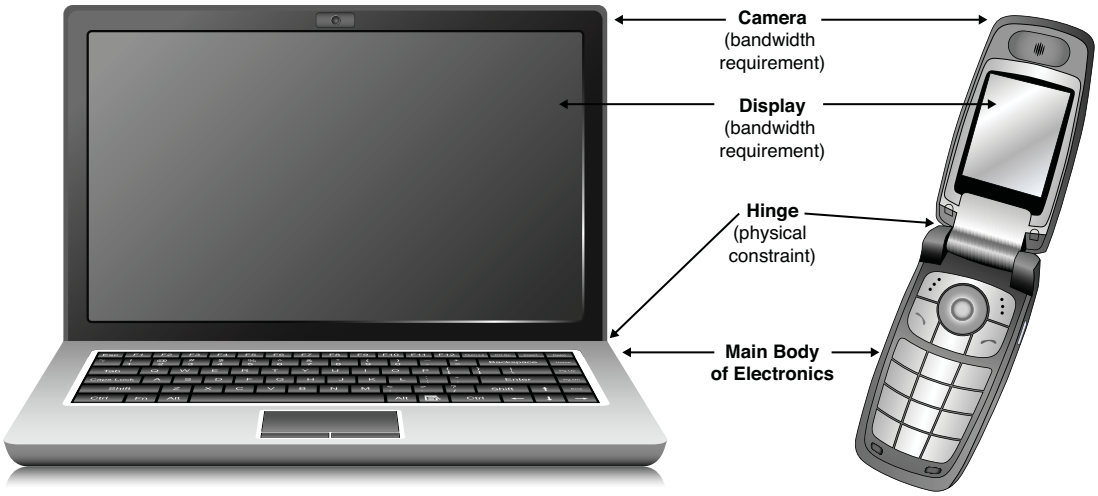
## Applications

Applications that benefit most from differential I/O are those with high bandwidth communication requirements such as graphic displays, cameras and imagers, or chip-to-chip interfaces.

While driving such displays using single-ended LVCMOS I/O is possible, portable or hand-held applications place additional physical constraints on a design, as shown in Figure 19-10. Typically, the high bandwidth device—the graphic display or camera—is separate from the main body that holds the majority of the system electronics. The main body and the high-bandwidth device are often mechanically connected by some sort of hinge mechanism. In other applications, the display and camera or cameras are folded into a compact phone, camera, or tablet body. Sending a wide, LVCMOS signal cable bundle across the hinge to the display is simply impractical. Likewise, a custom, wide, flex-cable is prohibitively expensive.

The higher bandwidth possible with differential signaling allows the same data to be transported over fewer electrical connections. Few connections results in a smaller, lower-cost flexible cable. Likewise, the smaller voltage swing results in lower electromagnetic interference (EMI).

***Figure 19-10. Portable Devices Benefit from Differential I/O***




iCE40 FPGAs offer a broad range of possible solutions for handheld applications, primarily in bridging and format conversion applications.

- Covert RGB data to high-speed differential data and back.
- Connect a processor without an integrated differential interface to a differential display or camera.
- Convert from one high-speed differential interface to another.
- Scale, rotate, and rebroadcast one stream onto another display format or another differential I/O format.

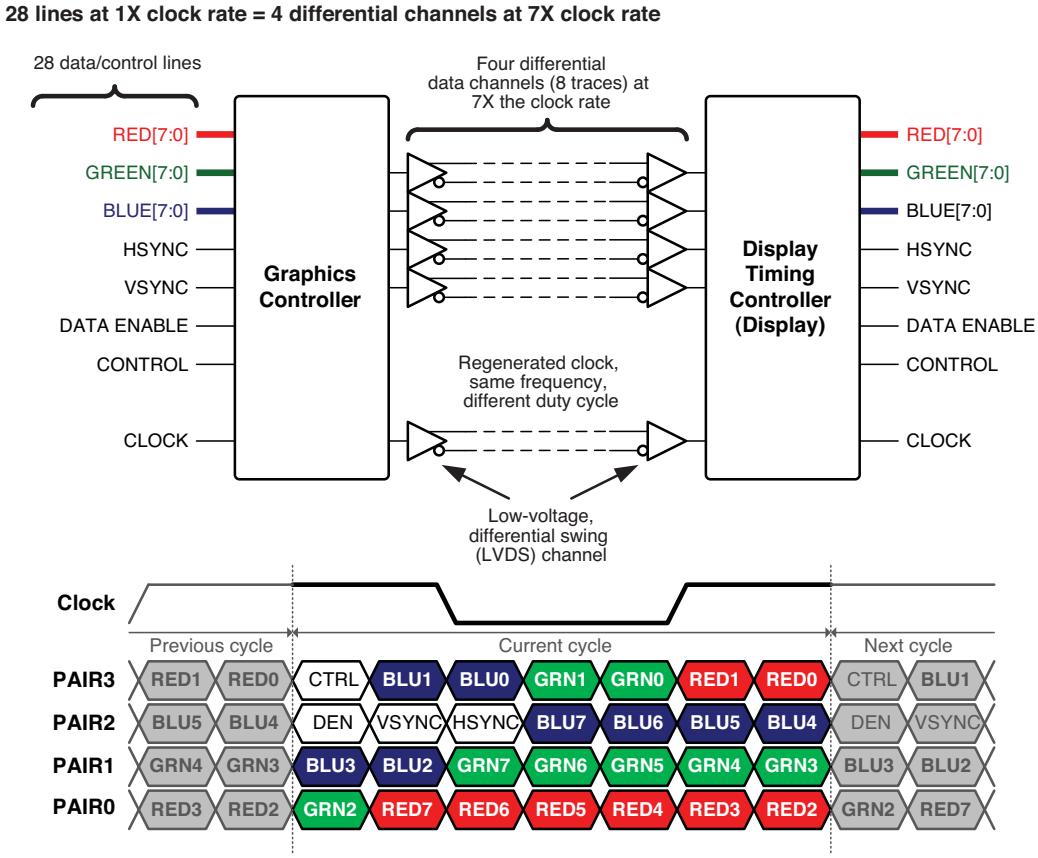
### Graphic Displays

Graphic displays demand high data rates, especially high-resolution displays that support a broad color range. In portable or handheld applications, the challenge is to provide a high-bandwidth communications path between the graphics controller and the display that fits with the physical constraints of the hinge or the package body. There are a variety of standard interfaces that leverage differential switching. Perhaps the most widely-used example is Flat Panel Display Link (FDP-Link), shown in Figure 19-11. The example shows a 24-bit per pixel design, although there are other implementations that use fewer colors and differential pairs. A standard 24-bit RGB interface

requires up to 28 single-ended signals. Instead of sending a cable bundle with 24 wires across the hinge, FPD-Link serializes the 28 data/control lines onto four differential I/O pairs, plus a clock differential pair resulting in 64% fewer wires. FPD-Link uses the Low-Voltage Differential Swing (LVDS) I/O standard.

Dividing 28 lines by four differential pairs also means that the data rate across the interface is seven times higher than the output clock rate.

***Figure 19-11. Example Differential I/O Solution: Flat Panel Display Interface***




At the receiving end, the differential data is de-serialized and converted back into a wider bundle of single-ended signals.

The integrated PLL in iCE40-family FPGAs simplifies this style of interface. An iCE40 FPGA can be at either end of the serial interface, either to allow a processor without an FPD-Link interface to communicate with an FPD-Link display, or to allow a processor with only an FPD-Link display interface to communicate to an RGB display or a display that uses a different format.

Figure 19-12 shows an example application running on an iCEman40 evaluation board. A 5-Megapixel camera captures live video images and provides the data to the iCE40 FPGA in RGGB format on a parallel LVTLL interface.

*Figure 19-12. 5-Megapixel CMOS Camera to LVDS Display on iCEman40 Evaluation Board*

### Cameras and Imagers

Differential interfaces are also popular for high-resolution and high-speed cameras and imagers.

## Summary

This technical note provides an overview of iCE LVDS technology, focusing on its advantages, implementation, application, and its various electrical and timing characteristics. It also includes detailed recommendations for instantiating LVDS transmitter and receivers in your design and calculating the required external terminations to guarantee optimum performance.

## References


- IEEE 1596.3 Standards
- Texas Instruments, "LVDS Owner's Manual", 2008
- Jimmy Ma, "A Closer Look at LVDS Technology", Pericom, 2001
- Texas Instruments, "Application Note 1032: An Introduction to FPD-Link"
- Texas Instruments, "Application Note 1127: LVDS Display Interface (LDI) TFT Data Mapping for Interoperability with FPD-Link"


## Technical Support Assistance


e-mail: techsupport@latticesemi.com

Internet: www.latticesemi.com

## Revision History

| Date | Version | Change Summary |
|---|---|---|
| September 2012 | 01.0 | Initial release. |
| March 2013 | 01.1 | Removed the Representative Electrical Characteristics of Various Differential I/O Standards table. |
| | | Updated information in Differential Outputs. |
| | | Updated description of figure showing 5-Megapixel CMOS Camera to LVDS Display on iCEman40 Evaluation Board. |
| | | Removed the Example FPD-Link Transmitter on iCEman40 Evaluation Board figure and description. |
| June 2013 | 01.2 | Updated the Differential Signaling Electrical Parameters and the iCE40 LVDS or Sub-LVDS Differential I/O Channel figures. |
| | | Updated the descriptions of the INPUT_CLOCK and the OUTPUT_CLOCK ports in the Port Names, Signal Direction, and Description for SB_IO (SB_GB_IO) Primitive table. |
| | | Updated Technical Support Assistance information. |
| October 2013 | 01.3 | Changed document title to “Using Differential I/O (LVDS, Sub-LVDS) in iCE40 LP/HX Devices”. |

# iCE40LM On-Chip Strobe Generator Usage Guide

October 2013 Technical Note TN1275

## Introduction

The iCE40LM family is an ultra-low power FPGA and sensor manager designed for ultra-low power mobile applications, such as smartphones, tablets and handheld devices. The iCE40LM family of devices includes integrated SPI and $I^2C$ blocks to interface with virtually all mobile sensors and application processors.

An ultra-low power 10KHz strobe generator is provided for Always-On applications and background polling that allow higher power processors to remain in power-down or sleep mode, conserving overall power consumption. A low power 12MHz strobe generator is provided for sensor management and pre-processing functions. These generators are intended for general clocking of internal logic and state machines.

### Key Features

Two strobe generators are available to users:

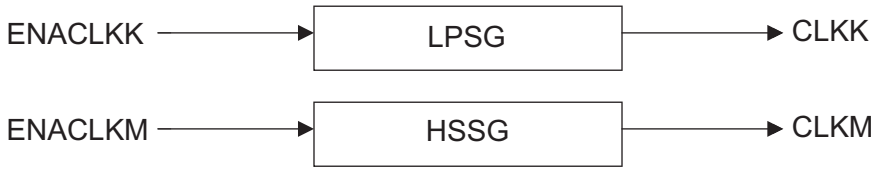
- LPSG – Low Power Strobe Generator
- HSSG – High Speed Strobe Generator

## On-Chip Strobe Generator Overview

You can access the two modules: LPSG and HSSG with enabled inputs and which you can dynamically control as shown in Figure 20-1.

LPSG runs at 10KHz and HSSG runs at 12MHz. LPSG and HSSG provide internal clock sources to user designs. These clocks can directly route to the global clock network or to local fabric.

***Figure 20-1. On-Chip Strobe Generator***




## I/O Port Description

***Table 20-1. LPSG I/O***

| Pin Name | Pin Direction | Description |
|---|---|---|
| ENACLKK | I | Enable LPSG |
| CLKK | O | LPSG Clock Output. |

***Table 20-2. HSSG I/O***

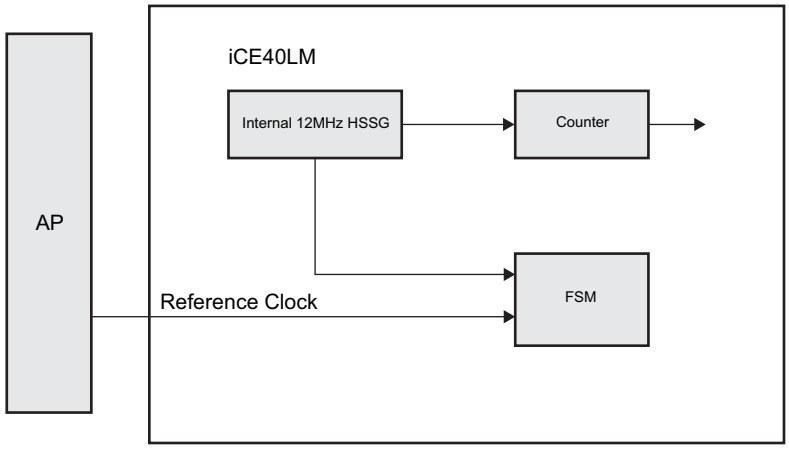
| Pin Name | Pin Direction | Description |
|---|---|---|
| ENACLKM | I | Enable HSSG |
| CLKM | O | HSSG Clock Output. |

## Connectivity Guideline

The LPSG and HSSG can be used as clock source. Their outputs are available for the user. They should be connected to the global clock network or local fabric. By default, the outputs will be routed to global clock network. To route to local fabric, please see the examples in the Appendix.

Note that Strobe Generator cannot provide accurate frequency. For applications that require more accuracy, it is recommended to use calibration circuit to support the strobe generator used as clock source. Figure 20-2 shows an example of the use of a reference clock that is only temporarily available for calibration.

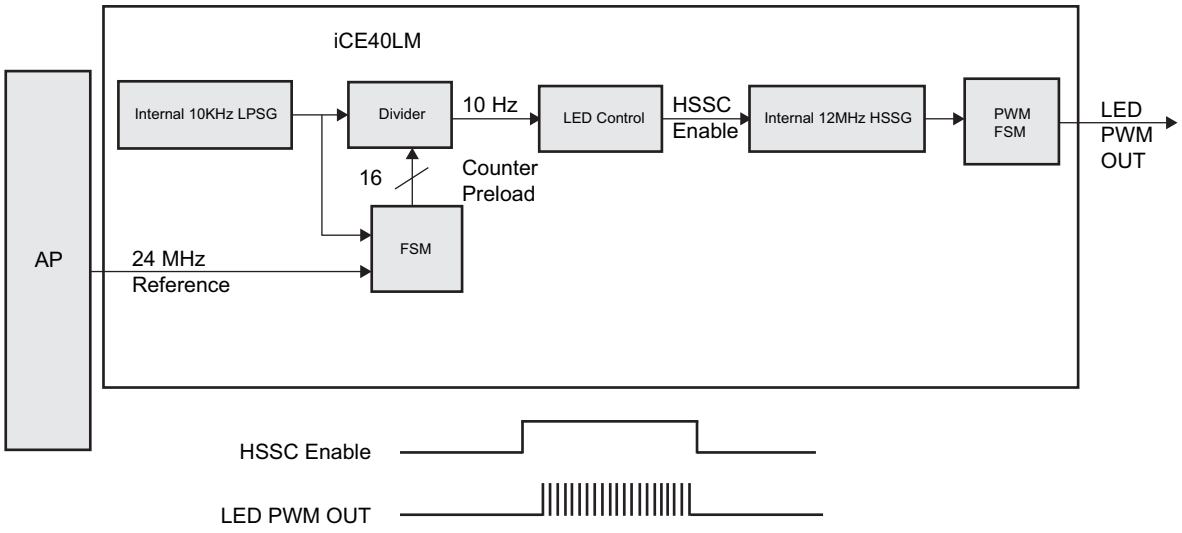
***Figure 20-2. Strobe Generator Calibration Example***




The calibration circuit for strobe generator can be improved for the purpose of power saving as shown in Figure 20-3.

In this example, 10KHz Strobe Generator is always on. Calibrated divider provides timing for LED on-off. When LED is on, LPSG Enable turns on 12MHz Strobe Generator (HSSG turns on in two cycles). PWM FSM provides accurate PWM for LED. Power benefit is12MHz only when LED is on and minimum power when LED is off

***Figure 20-3. Strobe Generator Used for Dynamic Clock Calibration That Can Be Used On Service LED***



## Power Management Options

When disabled, the LPSG and HSSG are in standby mode be default and consume only DC leakage. It is suggested to always enable LPSG and enable HSSG after there is an activity detected and the products return to full power mode for data analysis/processing.

## Technical Support Assistance

e-mail: techsupport@latticesemi.com

Internet: www.latticesemi.com

## Revision History

| Date | Version | Change Summary |
|---|---|---|
| October 2013 | 01.0 | Initial release. |

# Appendix: Design Entry

The following examples illustrate LPSG and HSSG usage with VHDL and Verilog.

## LSOSC (LPSG) Usage with VHDL

```
COMPONENT LSOSC
PORT (
  ENACLKK    : IN  std_logic;
  CLKK       : OUT std_logic);
END COMPONENT;

begin
OSCInst0: LSOSC
PORT MAP (ENACLKK    => ENACLKK,
            CLKK     => CLKK);
```

## LSOSC(LPSG) Usage with Verilog

```
module LSOSC(ENACLKK, CLKK);

input  ENACLKK;
output CLKK;

LSOSC OSCInst0 (.ENACLKK(ENACLKK),.CLKK(CLKK));

Endmodule
```

## HSOSC(HSSG) Usage with VHDL

```
COMPONENT HSOSC
PORT (
  ENACLKM    : IN  std_logic;
  CLKM       : OUT std_logic);
END COMPONENT;

begin
OSCInst0: HSOSC
PORT MAP (ENACLKM   => ENACLKM,
            CLKM     => CLKM);
```

## HSOSC(HSSG) Usage with Verilog (route through fabric)

```
module HSOSC(ENACLKM, CLKM);

input ENACLKM;
output CLKM;

HSOSC OSCInst0 (.ENACLKM(ENACLKM),.CLKM(CLKM))/* synthesis ROUTE_THROUGH_FABRIC=1 */;

endmodule
```

# iCE40LM SPI/I2C Hardened IP Usage Guide

October 2013 Technical Note TN1274

## Introduction

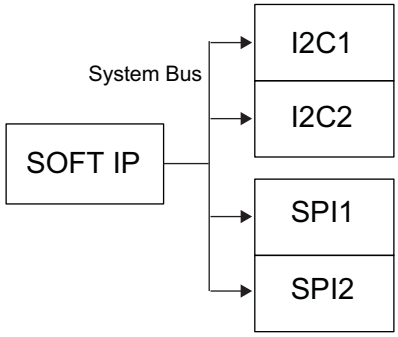
The iCE40LM family is an ultra-low power FPGA and sensor manager designed for ultra-low power mobile applications, such as smartphones, tablets and hand-held devices. The iCE40LM family of devices includes integrated SPI and I2C blocks to interface with virtually all mobile sensors and application processors.

The key components available to users are:

- Two I2C IP cores located at the upper left corner and upper right corner of the chip
- Two SPI IP cores located at lower left corner and lower right corner of the chip

The iCE40LM uses a System Bus to connect its Hard IP to the fabric. The iCE40LM device does not preload the Hard IP registers during configuration, thus, a soft IP is required. Using Module Generator to generate the IP is recommended because Module Generator generates the corresponding soft IP as well. For details, refer to “Module Generator” section.

***Figure 21-1. IP Block Diagram***




## I2C IP Core Overview

The I2C hard IP provides industry standard two pin communication interface that conforms to V2.1 of the I2C bus specification. It could be configured as either master or slave port. In master mode, it supports configurable data transfer rate and performs arbitration detection to allow it to operate in multi-master systems. It supports clock stretching in both master and slave modes with enable/disable capability. It supports both 7 bits and 10 bits addressing in slave mode with configurable slave address. It supports general call address detection in both master and slave mode. It provides interrupt logic for easy communicating with host. It also provides configurable digital delay at SDA output for reliably generating start/stop condition.

### Key Features

- Configurable Master and Slave mode
- Support for 7-bit or 10-bit configurable Slave Address
- Multi-master Arbitration Support
- Clock stretching to ensure data setup time
- Up to 400 kHz Data Transfer Speed; also support 100 kHz, 50 kHz modes
- General Call Support

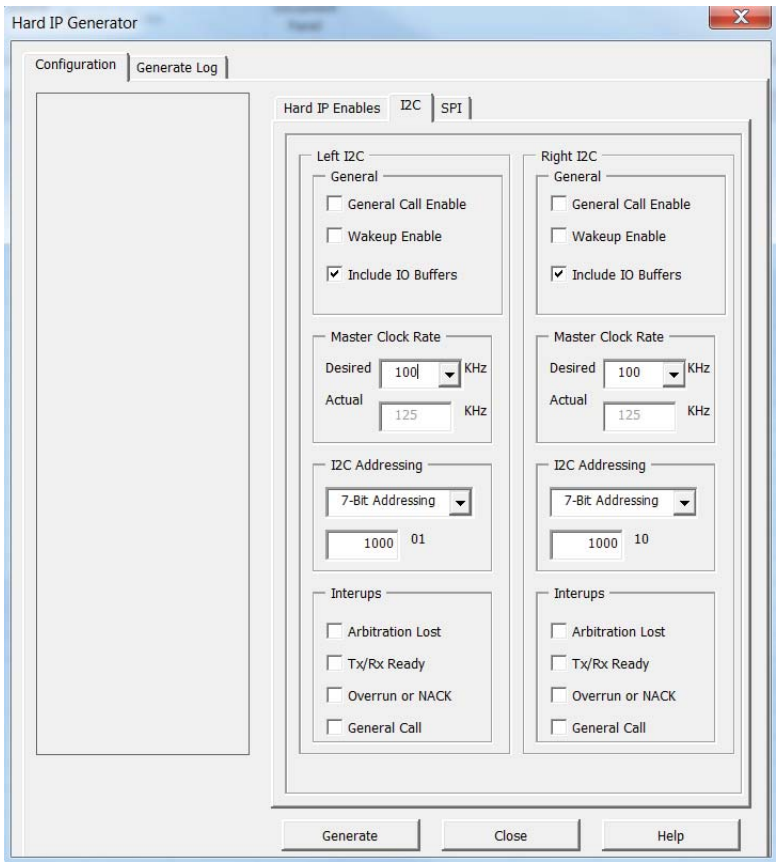
- Master clock source from System Bus clock
- Communication with custom logic through 8-bit wide data bus
- Optional dedicated pin location (SCL & SDA) for the I2C port
- Programmable 5 MSB bits for 7 bits Slave Address or 8 MSB bits for 10 bits Slave Address for user logic Slave I2C port.
- I2C port and all System Bus addressable registers are reset upon Power On Reset (POR)
- 30ns analog delay required at SDA input for reliable START, STOP condition detection
- Interface to customer logic through the System Bus Interface

## I2C Usage with Module Generation

The Module generator is the recommended flow for using the I2C Hard IP block. Please see section of “Module Generation” for more information.

When the I2C portion of the Hard IP block is disabled, all settings on the I2C tab shall be disabled.

***Figure 21-2. I2C Module GUI***




**General Call Enable**
This setting enables the I2C General Call response (addresses all devices on the bus using the I2C address 0) in Slave mode. This setting can be modified dynamically by enabling the GCEN bit in the I2C Control Register I2CCR1.

**Wakeup Enable**
Turns on the I2C wakeup on address match. Enables the Wakeup port. The WKUPEN bit in the I2CCR1 can be modified dynamically allowing the Wake Up function to be enabled or disabled.

**Master Clock (Desired)**
This edit box allows the user to specify a desired master clock frequency. A calculation is then made to determine a divider value to generate a clock close to this value from the input clock. The frequency of the input System Bus clock is specified on the main/general tab. The divider value is rounded to the nearest integer after dividing the input System Bus clock by the value entered in this field.

**Master Clock (Actual)**
Since it is not always possible to divide the input System Bus clock to the exact value requested by the user, the actual value will be returned in this read-only field. When both the desired I2C clock and System Bus clock fields contain valid data and either is updated, the Master Clock field returns the value (FREQ_SB / L2C_CLK_DIVIDER), rounded to an integer as shown in the example below. FREQ_SB is the System Bus Clock Frequency customer will enter in the Module Generator. I2C_CLK_DIVIDER will be a factor to be calculated as in the example below.

Master Clock Calculation Example:

1. Divider = I2C_CLK_DIVIDER = FREQ_SB / I2C_CLK_FREQ
2. Master Clock (Actual) = FREQ_SB / DividerInteger

For example: FREQ_SB = 42.5 MHz, I2C_BUS_PERF = 400 KHZ
1.Divider = I2C_CLK_DIVIDER = FREQ_SB / I2C_BUS_PERF
Divider = 42500 / 400 = 106.25
Therefore ROUND DividerInteger = 106
2.Master Clock (Actual) = FREQ_SB / DividerInteger
Actual frequency = 42500 / 106 = 400.9 KHz

In this example, if the user sees that an I2C_CLK_FREQ of 400 KHz cannot be generated, the user may choose to use the actual value or change the System Bus Clock. For this reason, the user needs to enter a frequency and the actual frequency will be displayed. This value may then be adjusted it desired.

**I2C Addressing**
This option allows the user to set 7-bit or 10-bit addressing and define the Hard I2C address.

**Interrupts**
When any of the interrupts are enabled, the I2C port is also enabled.

Arbitration Lost Interrupts: An interrupt which indicates I2C lost arbitration. This interrupt is bit IRQARBL of the register I2CIRQ. When enabled, it indicates that ARBL is asserted. Writing a '1' to this bit clears the interrupt. This option can be changed dynamically by modifying the bit IRQARBLEN in the register I2CIRQEN.

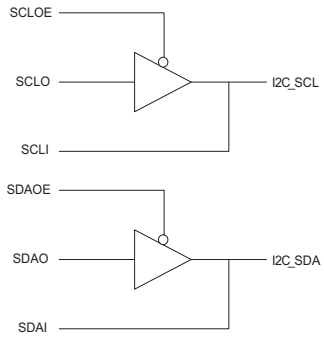
TX/RX Ready: An interrupt which indicates that the I2C transmit data register (I2CTXDR) is empty or that the receive data register (I2CRXDR) is full. The interrupt bit is IRQTRRDY of the register I2CIRQ. When enabled, it indicates that TRRDY is asserted. Writing a '1' to this bit clears the interrupt. This option can be changed dynamically by modifying the bit IRQTRRDYEN in the register I2CIRQEN.

Overrun or NACK: An interrupt which indicates that the I2CRXDR received new data before the previous data. The interrupt is bit IRQROE of the register I2CIRQ. When enabled, it indicates that ROE is asserted. Writing a '1' to this bit clears the interrupt. This option can be changed dynamically by modifying the bit IRQROEEN in the register I2CIRQEN.

General Call Interrupts: An interrupt which indicates that a general call has occurred. The interrupt is bit IRQHGC of the register I2CIRQ. When enabled, it indicates that ROE is asserted. Writing a '1' to this bit clears the interrupt. This option can be changed dynamically by modifying the bit IRQHGCEN in the register I2CIRQEN.

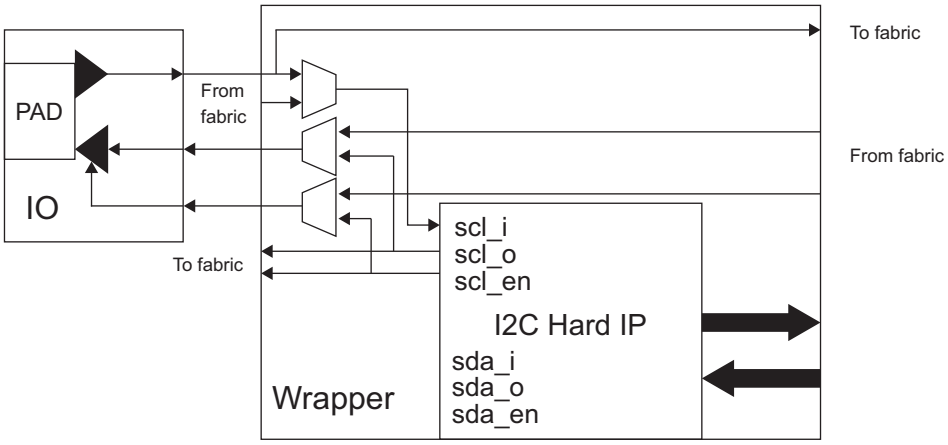
**Include IO Buffers**
Includes buffers for the I2C pins.

***Figure 21-3. I2C IO Buffers***




To use the dedicated I2C pins, start by instantiating or inferring the IO buffer. The I2C_SCL or I2C_SDA are then assigned to the dedicated pins unless the dedicated pins are not available. Note that the I2C pins do not have to use the dedicated pins. Refer to Figure 21-4 for the routing connections.

***Figure 21-4. I2C IO Connections***



## SB_I2C Hard IP Macro Ports and Wrapper Connections

When the I2C Hard IP is enabled, the necessary signals are included in the generated module.

***Table 21-1. Pins for the Hard I2C IP***

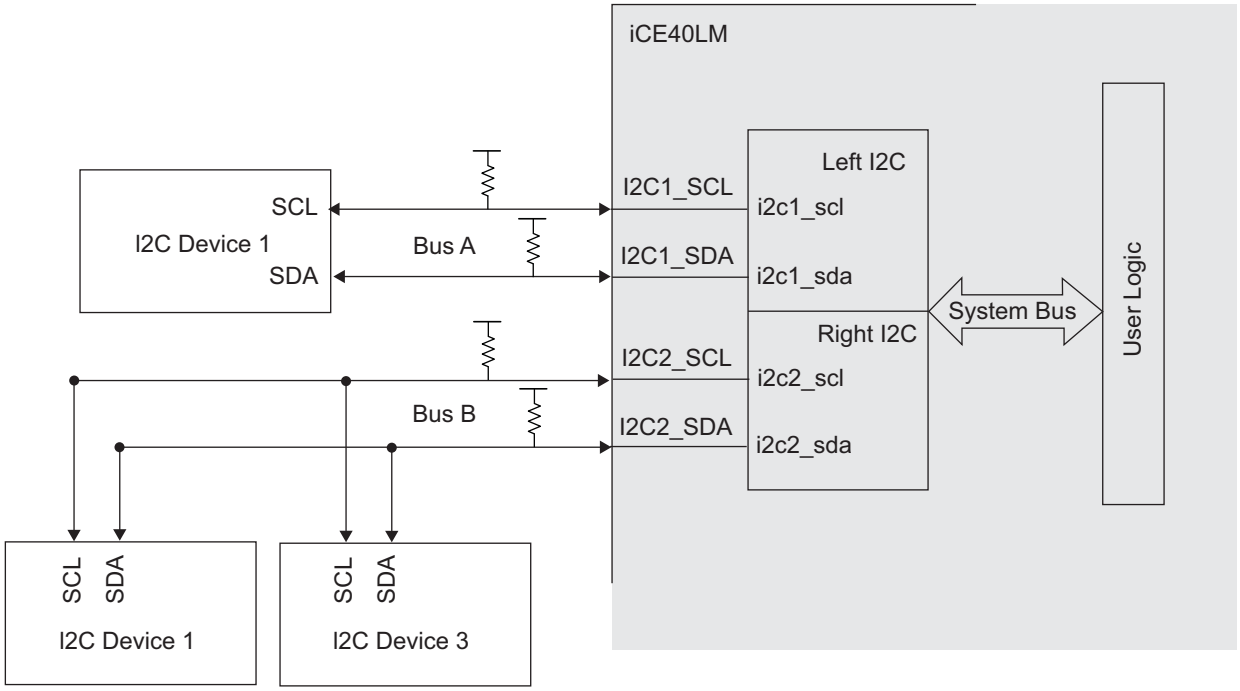
| Pin Name | Pin Direction | Description |
|---|---|---|
| SBCLKI | I | System clock input |
| SBRWI | O | System Read/Write input. R=0, W=1 |
| SBSTBI | O | System Strobe Signal |
| SBADRI[7:0] | I | System Bus Control Registers Address |
| SBDATI[7:0] | I | System Data Input |
| SBDATO[7:0] | O | System Data Output |
| SBACKO | O | System Acknowledgement |
| I2C_IRQO | O | Interrupt request output signal of the I2C core – The intended use of this signal is for it to be connected to a Master controller (such as a microcontroller or state machine) and request an interrupt when a specific condition is met. |
| I2C_WAKE | O | Wake-up signal –The signal is enabled only if the Wakeup Enable feature is set. |
| SCL | BI-directional | Open drain clock line of the I2C core – The signal is an output if the I2C core is performing a Master operation. The signal is an input for Slave operations. This signal can use a dedicated I/O pin. |
| SDA | Bi-directional | Open drain data line of the I2C core – The signal is an output when data is transmitted from the I2C core. The signal is an input when data is received into the I2C core. This signal can use dedicated I/O pin. |

## iCE40LM I2C Usage Cases

The I2C usage cases described below refer to Figure 21-5.

1. Master iCE40LM I2C Accessing Slave External I2C Devices
   a. A System Bus Master is implemented in the iCE40LM logic
   b. I2C devices 1, 2, and 3 are all Slave devicesc.The external SPI Master does not have access to the MachXO2 Configuration Logic because the SN that selects the Configuration Logic is pulled high.
   c. The Master performs bus transactions to the left I2C controller to access external Slave I2C Device 1 on Bus A.
   d. The Master performs bus transactions to the right I2C controller to access the external Slave I2C Devices number 2 or 3 on Bus B.
2. External Master I2C Device Accessing Slave iCE40LM I2C
   a. The I2C devices 1, 2, and 3 are I2C Master devices
   b. The external master I2C Device 1 on Bus A performs I2C memory cycles to access the left I2C controller using address yyyxxxxx01.
   c. The external master I2C Device 2 or 3 on Bus B performs I2C memory cycles to access the right I2C User with the address yyyxxxxx10
   d. A Master in the iCE40LM fabric must manage data reception and transmission. The Master can use interrupts or polling techniques to manage data transfer, and to prevent data overrun conditions.

***Figure 21-5. I2C Circuit***



## SPI IP Core Overview

The SPI hard IP provide industry standard four-pin communication interface with 8 bit wide System Bus to communicate with System Host. It could be configured as Master or Slave SPI port with separate Chip Select Pin. In master mode, it provides programmable baud rate, and supports CS HOLD capability for multiple transfers. It provides variety status flags, such as Mode Fault Error flag, Transmit/Receive status flag etc. for easy communicate with system host.

### Key Features

Configurable Master and slave mode

- Full-Duplex data transfer
- Mode fault error flag with CPU interrupt capability
- Double-buffered data register
- Serial clock with programmable polarity and phase
- LSB First or MSB First Data Transfer
- Communicate with custom logic through 8 bit wide System Bus
- Optional dedicated pin location for (SCK, MOSI, MISO) for SPI port.
- Maximum 4 Slave select output could be routed to any IO through fabric for Master mode
- Slave chip select pin could be routed to any IO through fabric for custom logic
- Clock source for Master clock (MCLK) generation come from System Bus clock.
- Optionally send out status byte to inform remote SPI master the slave receiving register status during slave write.
- Optionally enable the Slow Response Mode for slave SPI read, which will automatically deploy the Lattice specific protocol to resolve the issue that caused by the slow initial respond of the System host at high SPI clock rate.
- The SPI port and all System Bus addressable registers will be reset upon Power On Reset (POR)
- Interface to customer logic through the System Bus Interface

## SPI Usage with Module Generation

The Module generator is the recommended flow for using the SPI Hard IP block. See section of “Module Generation” for more information.

When the SPI portion of the Hard IP block is disabled, all settings on the SPI tab shall be disabled.

***Figure 21-6. SPI Module GUI***


**SPI Mode (Enable Slave Interface)**
This option allows the user to enable Slave Mode interface for the initial state of the SPI block. By default, Slave Mode interface is enabled. Options and ports that are applicable to only one mode will be disabled when the other is chosen.

**SPI Mode (Enable Master Interface)**
This option allows the user to enable Master Mode interface for the initial state of the SPI block. This option can be updated dynamically by modifying the MSTR bit of the register SPICR2. Options and ports that are applicable to only one mode will be disabled when the other is chosen.

**Master Clock Rate (Desired)**
(“Enable Master Interface” only) This edit box allows the user to specify a desired master clock frequency. A calculation is then made to determine a divider value to generate a clock close to this value from the input clock. The frequency of the input System Bus clock is specified on the main/general tab. The divider value is rounded to the nearest integer after dividing the input System Bus clock by the value entered in this field.

**Master Clock Rate (Actual)**
("Enable Master Interface" only) Since it is not always possible to divide the input System Bus clock exactly to that requested by the user, the actual value will be returned in this read-only field. When both the desired SPI clock and System Bus clock fields have valid data and either is updated, this field returns the value (FREQ_SB / SPI_CLK_DIVIDER), rounded to two decimal places as shown in the example below. FREQ_SB is the System Bus Clock Frequency that the customer enters in the Module Generator. SPI_CLK_DIVIDER is a factor to be calculated, which is also shown in the example below.

Master Clock Calculation Example:

1. Divider = SPI_CLK_DIVIDER = FREQ_SB / SPI_CLK_FREQ
2. Master Clock (Actual) = FREQ_SB / DividerInteger

For example: FREQ_SB = 66 MHz, SPI_CLK_FREQ = 25 MHZ

1.Divider = SPI_CLK_DIVIDER = FREQ_SB / SPI_CLK_FREQ
Divider = 66 / 25 = 2.64
Therefore ROUND DividerInteger = 3

2.Master Clock (Actual) = FREQ_SB / DividerInteger
Actual frequency = 66 / 3 = 22 MHz

In this example, if the user sees that an SPI_CLK_FREQ of 25 MHz cannot be generated, the user may choose to use the actual value or change the System Bus Clock. For this reason, the user needs to enter a frequency and the actual frequency will be displayed. This value may then be adjusted it desired.

**LSB First**
This setting specifies the order of the serial shift of a byte of data. The data order (MSB or LSB first) is programmable within the SPI core. This option can be updated dynamically by modifying the LSBF bit in the register SPICR2.

**Inverted Clock**
The inverts the clock polarity used to sample and output data is programmable for the SPI core. When selected the edge changes from the rising to the falling clock edge. This option can be updated dynamically by accessing the CPOL bit of register SPICR2.

**Phase Adjust**
An alternate clock-data relationship is available for SPI devices with particular requirements. This option allows the user to specify a phase change to match the application. This option can be updated dynamically by accessing the CPHA bit in the register SPICR2.

**Slave Handshake Mode**
Enables Lattice proprietary extension to the SPI protocol. For use when the internal sup-port circuit (e.g. WISHBONE host) cannot respond with initial data within the time required, and to make the Slave read out data predictably available at high SPI clock rates. This option can be updated dynamically by accessing the SDBRE bit in the register SPICR2.

**Master Chip Selects**
("Enable Master Interface" only) The core has the ability to provide up to 4 individual chip select outputs for master operation. This field allows the user to prevent extra chip selects from being brought out of the core. This option can be updated dynamically by modifying the register SPICSR.

**SPI Controller Interrupts**
TX Ready: An interrupt which indicates the SPI transmit data register (SPITXDR) is empty. The interrupt bit is IRQTRDY of the register SPIIRQ. When enabled, indicates TRDY was asserted. Write a '1' to this bit to clear the interrupt. This option can be change dynamically by modifying the bit IRQTRDYEN in the register SPIIRQEN.

RX Ready: An interrupt which indicates the receive data register (SPIRXDR) contains valid receive data. The interrupt is bit IRQRRDY of the register SPIIRQ. When enabled, indicates RRDY was asserted. Write a '1' to this bit to clear the interrupt. This option can be change dynamically by modifying the bit IRQRRDYEN in the register SPICSR

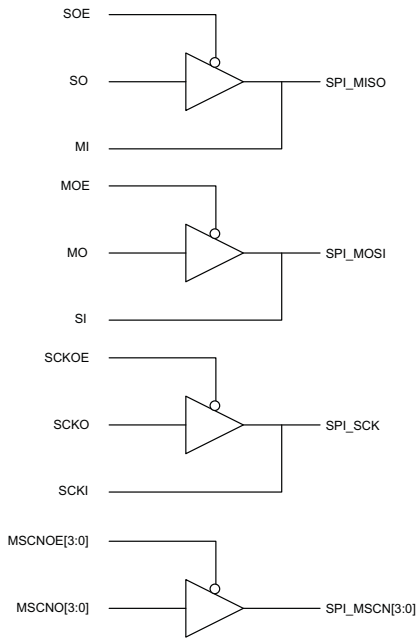
TX Overrun: An interrupt which indicates the Slave SPI chip select (SPI_SCSN) was driven low while a SPI Master. The interrupt is bit IRQMDF of the register SPIIRQ. When enabled, indicates MDF (Mode Fault) was asserted. Write a '1' to this bit to clear the interrupt. This option can be change dynamically by modifying the bit IRQMDFEN in the register SPIIRQEN.

RX Overrun: An interrupt which indicates SPIRXDR received new data before the previous data. The interrupt is bit IRQROE of the register SPIIRQ. When enabled, indicates ROE was asserted. Write a '1' to this bit to clear the interrupt. This option can be change dynamically by modifying the bit IRQROEEN in the register SPIIRQEN.

**Wakeup Enable**
The core can optionally provide a wakeup signal to the device to resume from low power mode. This option can be updated dynamically by modifying the bit WKUPEN_USER in the register SPICR1.

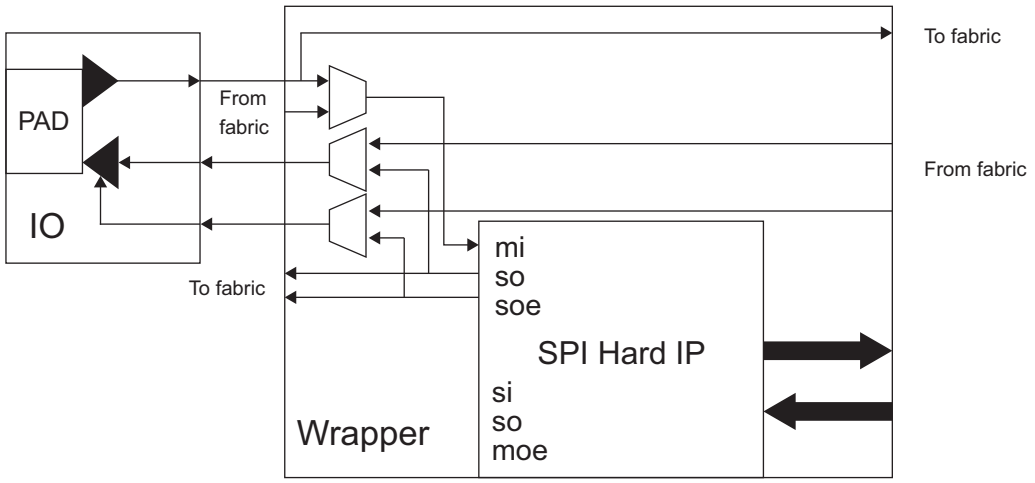
**Include IO Buffers**
Includes buffers for the SPI pins.

***Figure 21-7. SPI IO Buffers included***




To use the dedicated SPI pins the user would start by instantiating or inferring the IO buffer. The SPI_MISO, SPI_MOSI, SPI_SCK, SPI_MCSNO[0] would then be assigned to the dedicated pins unless the dedicated pins are not available. Note the SPI pins do not have to use the dedicated pins. Refer to the diagram below for the routing connections.

*Figure 21-8. SPI IO Connections*




## SB_SPI Hard IP Macro Ports and Wrapper Connections

When the SPI Hard IP is enabled, the necessary signals will be included in the generated module.

*Table 21-2. Pins for the Hard SPI IP*

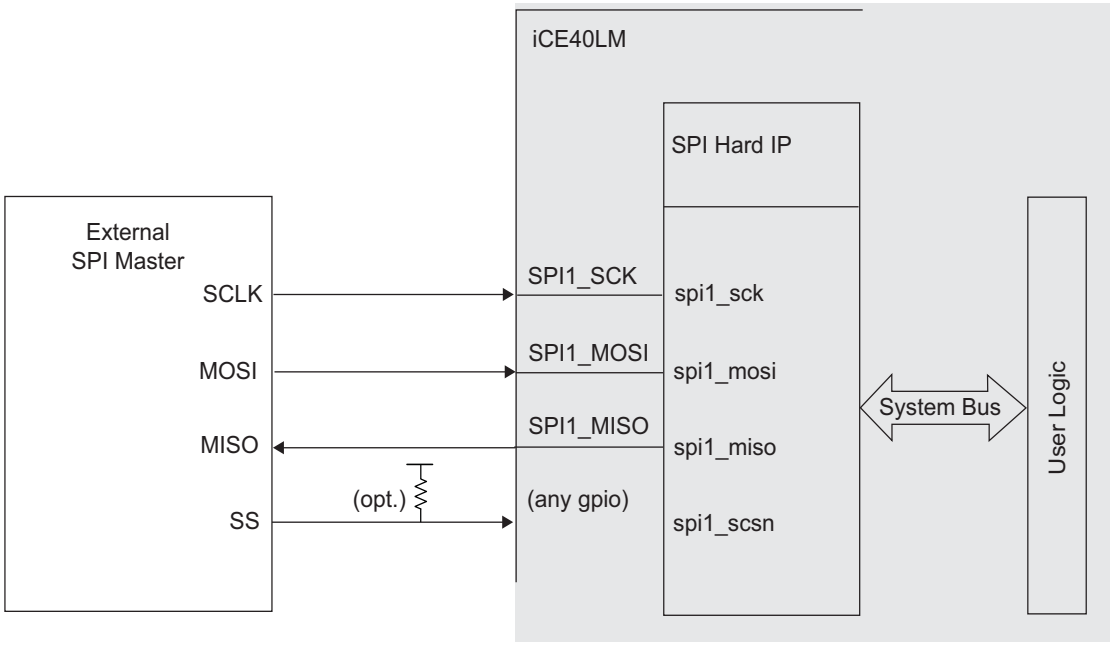
| Pin Name | Pin Direction | Description |
|---|---|---|
| SBCLKI | I | System clock input |
| SBRWI | O | System Read/Write input. R=0, W=1 |
| SBSTBI | O | System Strobe Signal |
| SBADRI[7:0] | I | System Bus Control Registers Address |
| SBDATI[7:0] | I | System Data Input |
| SBDATO[7:0] | O | System Data Output |
| SBACKO | O | System Acknowledgement |
| SPI_IRQO | O | Interrupt request output signal of the I2C core – The intended use of this signal is for it to be connected to a Master controller (i.e. a microcontroller or state machine) and request an interrupt when a specific condition is met. |
| SPI_WAKE | O | Wake-up signal – The signal is enabled only if the Wakeup Enable feature has been set |
| SPI_SCK | BI-directional | The signal is an output if the SPI core is in Master mode (MCLK). The signal is an input if the SPI core is in Slave mode (CCLK). This signal could use dedicated I/O pin. |
| SPI_MOSI | Bi-directional | The signal is an output if the SPI core is in Master mode (SISPI). The signal is an input if the SPI core is in Slave mode (SI). This signal could use dedicated I/O pin. |
| SPI_MISO | BI-directional | The signal is an input if the SPI core is in Master mode (SPISO). The signal is an output if the SPI core is in Slave mode (SO). This signal could use dedicated I/O pin. |
| SPI_SCSNI | I | User Slave Chip Select (Active Low). An external SPI Master controller asserts this signal to transfer data to/from the SPI Controllers Transmit Data/Receive Data registers. This signal could use dedicated I/O pin. The dedicated pin is shared with SPI_NCSNO[0]. |
| SPI_MCSNO[3:0] | O | Master Chip Select (Active Low). Up to 4 independent slave SPI devices can be accessed using the SPI Controller when it is in Master SPI mode. Only SPI_MCSNO[0] could use dedicated I/O pin |

## ICE40LM SPI Usage Cases

The SPI usage cases described below refer to Figure 21-9 and Figure 21-10.

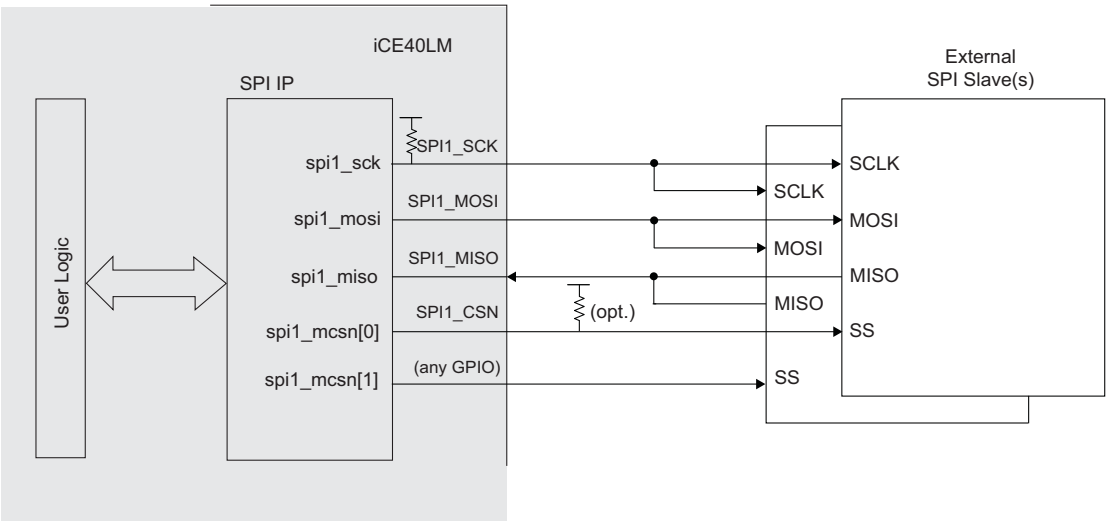
1. External Master SPI Device Accessing the Slave iCE40LM User SPI
   a. The External Master SPI is connected to the iCE40LM using the dedicated SI, SO, CCLK pins. The spi_scsn is placed on any Generic I/O. The SPI Mode is set to Slave only.
   b. A Master controller is implemented in the iCE40LM general purpose logic array. The master controller monitors the availability to transmit or receive data by polling the SPI status registers, or by responding to interrupts generated by the SPI Controller.

***Figure 21-9. External Master SPI Device Accessing the Slave User SPI***



2. iCE40LM User SPI Master accessing one or multiple External Slave SPI devices
   a. The iCE40LM SPI Master is connected to External SPI Slave devices. The Chip Selects are configured as follows:
      i. The iCE40LM SPI Master Chip Select spi_mcsn[0] is placed on the dedicated CSSPIN and connected to the External Slave Chip Select.
      ii. The iCE40LM SPI Master Chip Select spi_mcsn[1] is placed on any I/O and connected to another External Slave Chip Select.
      iii. Up to 4 External Slave SPIs can be connected using spi_mcsn[3:0]
   b. A Master controller is implemented in the iCE40LM general logic. It controls transfers to the slave SPI devices. It can use a polling method, or it can use SPI Controller interrupts to manage transfer and reception of data.

***Figure 21-10. iCE40LM User SPI Master Accessing One or Multiple External Slave SPI Devices***



# Module Generator

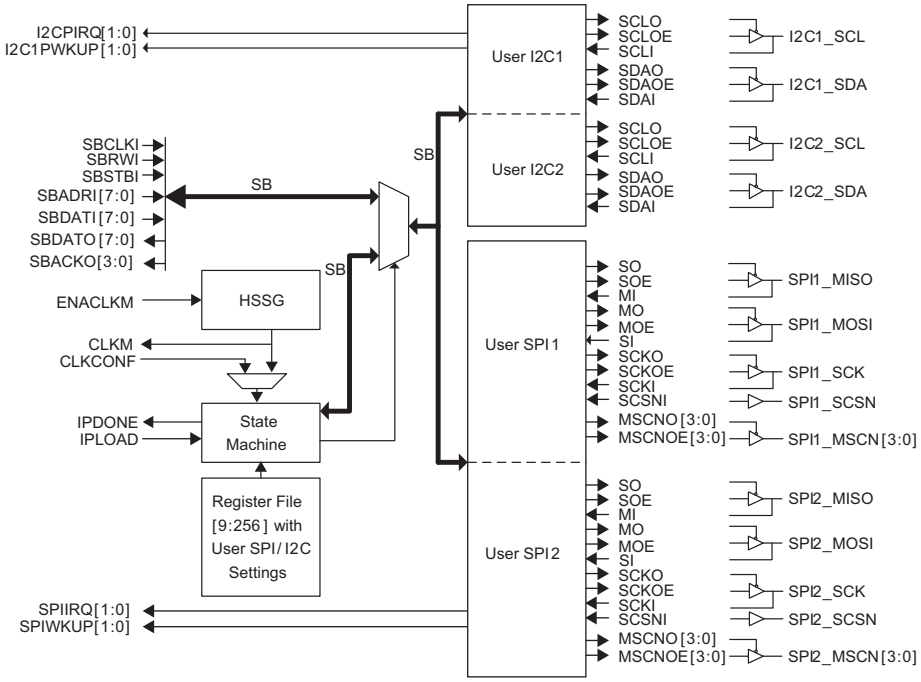
The iCE40LM uses a System Bus to connect its Hard IP to the fabric. The System bus is connected to a 2 SPI and 2 I2C Hard IPs. The iCE40LM device does not preload the Hard IP registers during configuration so a soft IP will be required.

## Key Features

- Module Generator will generate the Soft IP which is a wrapper around the Hard IP
- Soft IP will release the System Bus after the Hard IPs have been configured
- The Soft IP by default will use the HSSG. Details about HSSG please refer to TN1275, iCE40LM On-Chip Strobe Generator Usage Guide.
- The Register file will be [9:256]
  - The address size is 256 bits which encompasses all address of the Hard IPs
  - The data size is 9 bits with the format [W, Register Data]. When W=1 the state machine will write the corresponding Hard IP address with data

See TN1276, Advanced iCE40LM SPI/I2C Hardened IP Usage Guide for more information.

***Figure 21-11. Soft IP Block Diagram***

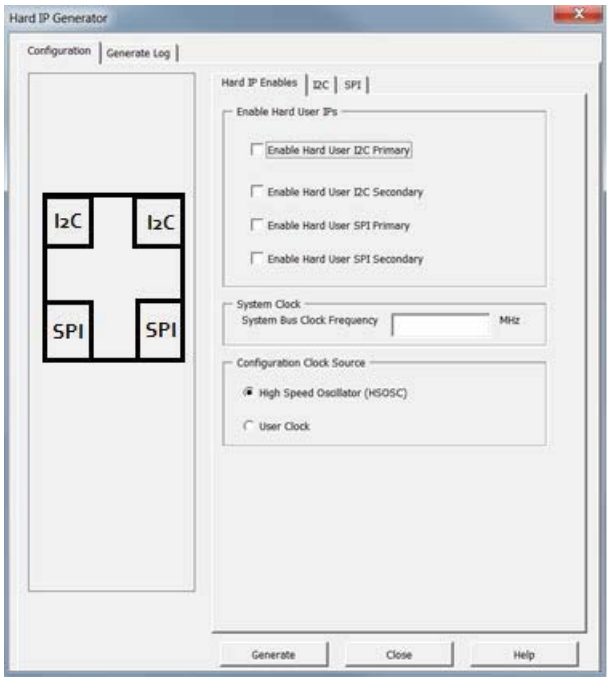



Notes: Only SPI*_MSCN[1:0] have dedicated pins

IPDONE = 1 when Hard IP Configuration is complete, IPDONE = 0 during configuration

IPLOAD is an input signal which is used to begin configuration on a positive edge

*Figure 21-12. Module Generator*




**Enable Left I2C**
This option allows the user to enable left I2C on the I2C Tab

**Enable Right I2C**
This option allows the user to enable right I2C on the I2C Tab

**Enable Left SPI**
This option allows the user to enable left SPI on the SPI Tab

**Enable Right SPI**
This option allows the user to enable right SPI on the SPI Tab

**System Clock**
User setting which is used to set the divider settings of the I2C and SPI

**Configuration Clock Source**
User will select to use the HSSG or the CLKCONF (User provided clock). details please refer to TN1275, iCE40LM On-Chip Strobe Generator Usage Guide for HSSG.

## Technical Support Assistance

e-mail: techsupport@latticesemi.com

Internet: www.latticesemi.com

## Revision History

| Date | Version | Change Summary |
|---|---|---|
| October 2013 | 01.0 | Initial release. |

# Advanced iCE40LM SPI/I2C Hardened IP Usage Guide

October 2013 Technical Note TN1276

## Introduction

This reference guide provides guidance for the advanced usage of iCE40LM SPI/I2C IP. Use it as a supplement to TN1274, iCE40LM SPI/I2C Hardened IP Usage Guide for more complete information. In this document you will find:

- System Bus Protocol
- I2C/SPI Register Mapping
- I2C/SPI Timing Diagram
- Command Sequences
- Examples

## System Bus Interface

The System Bus in the iCE40LM provides connectivity between FPGA user logic and the Hardened IP functional blocks. The user can implement a System Bus Master interface to interact with the Hardened IP System Bus Slave interface.

The block diagram in Figure 22-1 shows the supported System Bus signals between the FPGA core and the Hardened IP. Table 22-1 provides a detailed definition of the supported signals.

***Figure 22-1. System Bus Interface Between the FPGA Core and the IP***

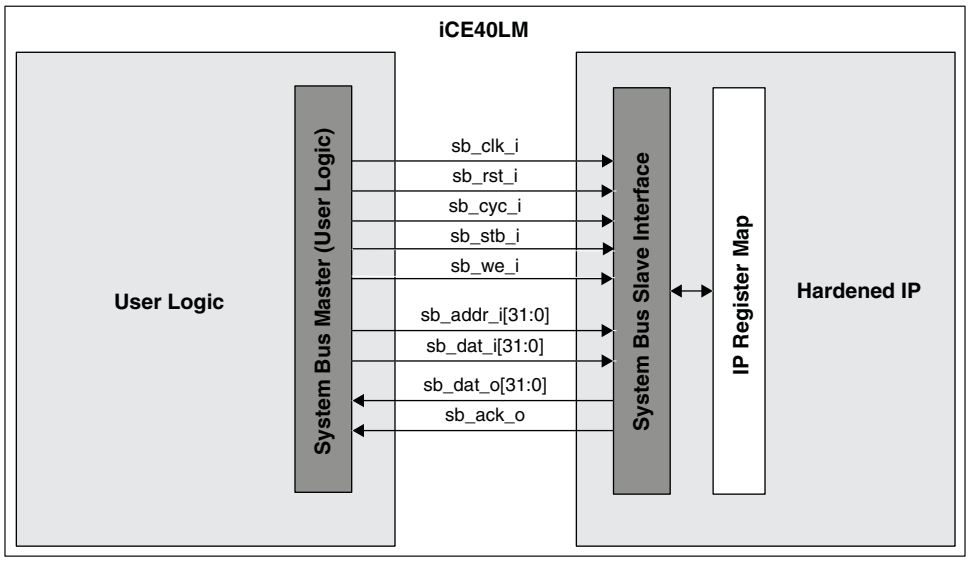


*Table 22-1. System Bus Slave Interface Signals of the Hardened IP Module*

| Signal Name | I/O | Width | Description |
|---|---|---|---|
| sb_clk_i | Input | 1 | Positive edge clock used by System Bus Interface registers and hardened functions. Supports clock speeds up to 133 MHz. |
| sb_rst_i | Input | 1 | Active-high, synchronous reset signal that will only reset the System Bus interface logic. This signal will not affect the contents of any registers. It will only affect ongoing bus transactions. Wait 1us after de-assertion before starting any subsequent System Bus transactions. |
| sb_cyc_i | Input | 1 | Active-high signal, asserted by the System Bus master, indicates a valid bus cycle is present on the bus. |
| sb_stb_i | Input | 1 | Active-high strobe, input signal, indicating the System Bus slave is the target for the current transaction on the bus. The IP asserts an acknowledgment in response to the assertion of the strobe. |
| sb_we_i | Input | 1 | Level sensitive Write/Read control signal. Low indicates a Read operation, and High indicates a Write operation. |
| sb_adr_i | Input | 8 | 8-bit wide address used to select a specific register from the register map of the IP. |
| sb_dat_i | Input | 8 | 8-bit input data path used to write a byte of data to a specific register in the register map of the IP. |
| sb_dat_o | Output | 8 | 8-bit output data path used to read a byte of data from a specific register in the register map of the IP. |
| sb_ack_o | Output | 1 | Active-high, transfer acknowledge signal asserted by the IP, indicating the requested transfer is acknowledged. |

To interface with the IP, you must create a System Bus Master controller in the User Logic. In a multiple-Master configuration, the System Bus Master outputs are multiplexed in a user-defined arbiter. If two Masters request the bus in the same cycle, only the outputs of the arbitration winner reach the Slave interface.

## System Bus Write Cycle

Figure 22-2 shows the waveform of a Write cycle from the perspective of the System Bus Slave interface. During a single Write cycle, only one byte of data is written to the IP block from the System Bus Master. A Write operation requires a minimum three clock cycles.

On clock Edge 0, the Master updates the address, data and asserts control signals. During this cycle:

- The Master updates the address on the sb_adr_i[7:0] address lines
- Updates the data that will be written to the IP block, sb_dat_i[7:0] data lines
- Asserts the write enable sb_we_i signal, indicating a write cycle
- Asserts the sb_cyc_i to indicate the start of the cycle
- Asserts the sb_stb_i, selecting a specific slave module

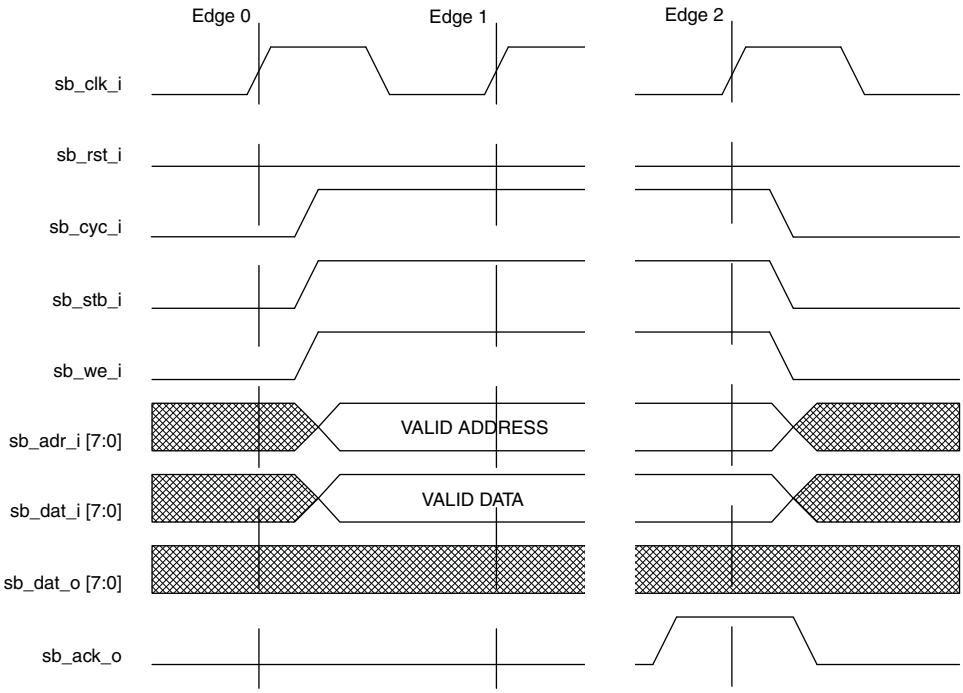
On clock Edge 1, the System Bus Slave decodes the input signals presented by the master. During this cycle:

- The Slave decodes the address presented on the sb_adr_i[7:0] address lines
- The Slave prepares to latch the data presented on the sb_dat_i[7:0] data lines
- The Master waits for an active-high level on the sb_ack_o line and prepares to terminate the cycle on the next clock edge, if an active-high level is detected on the sb_ack_o line
- The IP may insert wait states before asserting sb_ack_o, thereby allowing it to throttle the cycle speed. Any number of wait states may be added
- The Slave asserts sb_ack_o signal

The following occurs on clock Edge 2:

- The Slave latches the data presented on the sb_dat_i[7:0] data lines
- The Master de-asserts the strobe signal, sb_stb_i, the cycle signal, sb_cyc_i, and the write enable signal, sb_we_i
- The Slave de-asserts the acknowledge signal, sb_ack_o, in response to the Master de-assertion of the strobe signal

***Figure 22-2. System Bus Write Operation***




## System Bus Read Cycle

Figure 22-3 shows the waveform of a Read cycle from the perspective of the System Bus Slave interface. During a single Read cycle, only one byte of data is read from the IP block by the System Bus master. A Read operation requires a minimum three clock cycles.

On clock Edge 0, the Master updates the address, data and asserts control signals. The following occurs during this cycle:

- The Master updates the address on the sb_adr_i[7:0] address lines
- De-asserts the write enable sb_we_i signal, indicating a Read cycle
- Asserts the sb_cyc_i to indicate the start of the cycle
- Asserts the sb_stb_i, selecting a specific Slave module

On clock Edge 1, the System Bus slave decodes the input signals presented by the master. The following occurs during this cycle:

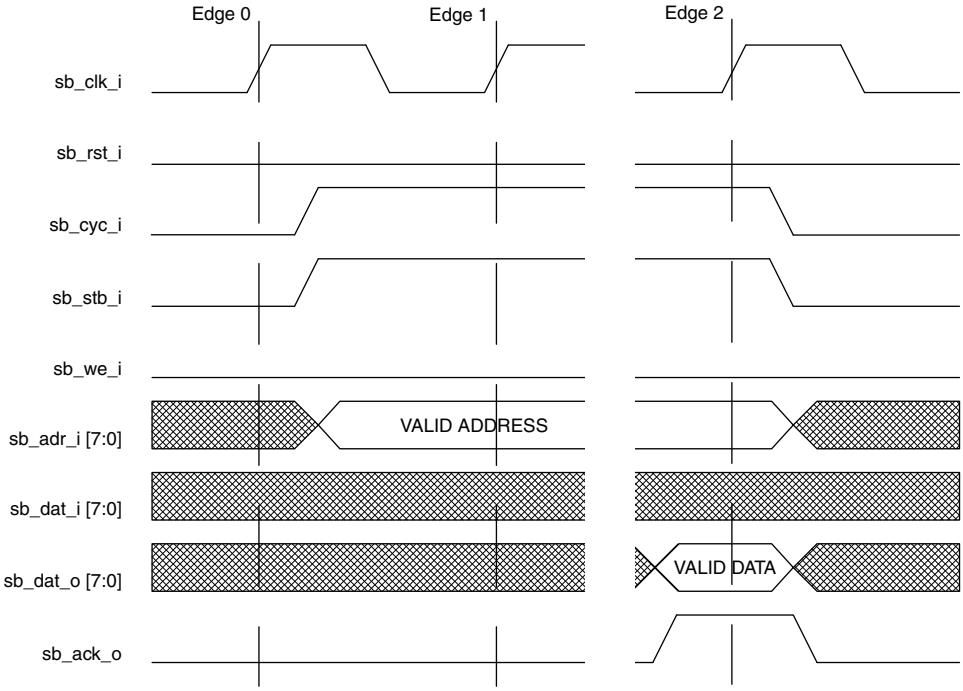
- The Slave decodes the address presented on the sb_adr_i[7:0] address lines
- The Master prepares to latch the data presented on sb_dat_o[7:0] data lines from the System Bus slave on the following clock edge
- The Master waits for an active-high level on the sb_ack_o line and prepares to terminate the cycle on the next clock edge, if an active-high level is detected on the sb_ack_o line

- The IP may insert wait states before asserting sb_ack_o, thereby allowing it to throttle the cycle speed. Any number of wait states may be added.
- The Slave presents valid data on the sb_dat_o[7:0] data lines
- The Slave asserts sb_ack_o signal in response to the strobe, sb_stb_i signal

The following occurs on clock Edge 2:

- The Master latches the data presented on the sb_dat_o[7:0] data lines
- The Master de-asserts the strobe signal, sb_stb_i, and the cycle signal, sb_cyc_i
- The Slave de-asserts the acknowledge signal, sb_ack_o, in response to the master de-assertion of the strobe signal

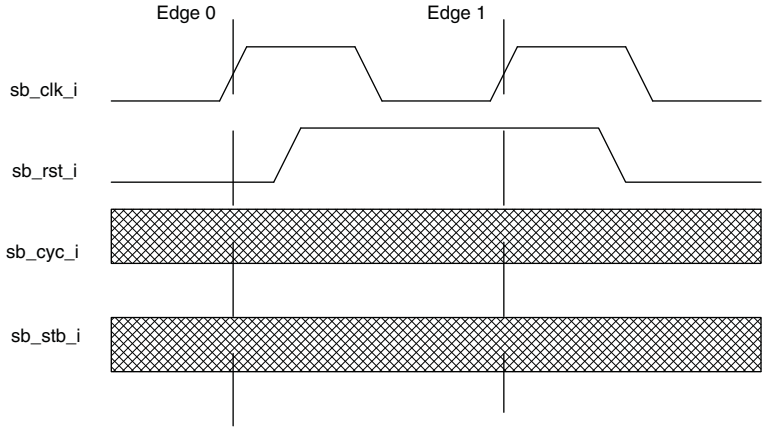
***Figure 22-3. System Bus Read Operation***



## System Bus Reset Cycle

Figure 22-4 shows the waveform of the synchronous sb_rst_i signal. Asserting the reset signal will only reset the System Bus interface logic. This signal will not affect the contents of any registers in the register map. It will only affect ongoing bus transactions.

***Figure 22-4. System Bus Interface Reset***




The sb_rst_i signal can be asserted for any length of time.

## Hardened I2C IP Cores

I2C is a widely used two-wire serial bus for communication between devices on the same board. Every iCE40LM device contains two hardened I2C IP cores. Either of the two cores can be operated as an I2C Master or as an I2C Slave.

## I2C Registers

Both I2C cores communicate with the System Bus interface through a set of control, command, status and data registers. Table 1 shows the register names and their functions.

***Table 22-2. I2C Registers Summary***

| I2C Register Name | Simulation Model Register Name | Address[3:0] | Register Function | Access |
|---|---|---|---|---|
| I2CCR1 | I2CCR1 | 1000 | Control | Read/Write |
| I2CCMDR | I2CCMDR | 1001 | Command | Read/Write |
| I2CSR | I2CSR | 1100 | Status | Read |
| I2CIRQ | I2CINTSR | 0110 | Interrupt Status | Read/Write[1] |
| I2CIRQEN | I2CINTCR | 0111 | Interrupt Enable | Read/Write |
| I2CGCDR | I2CGCDR | 1111 | General Call Information | Read |
| I2CSADDR | I2CSADDR | 0011 | Slave Address MSB | Read/Write |
| I2CTXDR | I2CTXDR | 1101 | Transmit Data | Write |
| I2CRXDR | I2CRXDR | 1110 | Receive Data | Read |
| I2CBRLSB | I2CBRLSB | 1010 | Clock Presale register, LSB | Read/Write |
| I2CBRMSB | I2CBRMSB | 1011 | Clock Presale register, MSB | Read/Write |

1. I2CIRQ is Read Only. Write operation upon this register will not change the content of this register, but will clear corresponding interrupt flag caused by the flags inside I2CIRQ.

*Table 22-3. I2C Control Register 1 (I2CCR1)*[1]

| I2CCR1 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | I2CEN | GCEN | WKUPEN | (Reserved) | SDA_DEL_SEL | | (Reserved) | (Reserved) |
| Default | 0 | 0 | 0 | 0 | 00 | | 0 | 0 |
| 0 to disable | YES | YES | YES | - | - | | - | - |
| Access | R/W | R/W | R/W | - | R/W | | - | - |

1. A write to this register will cause the I2C core to reset

I2CEN — I2C System Enable Bit – This bit enables the I2C core functions. If I2CEN is cleared, the 2C core is disabled and forced into idle state.

GCEN — Enable bit for General Call Response – Enables the general call response in slave mode.

0: Disable
1: Enable

The General Call address is defined as 0000000 and works with either 7-bit or 10-bit addressing

WKUPEN — Wake-up from Standby/Sleep(by Slave Address matching) Enable Bit – When this bit is enabled the, I2C core can send a wake-up signal to wake the device up from standby/sleep. The wake-up function is activated when the Slave Address is matched during standby/sleep mode.

SDA_DEL_SEL[1:0] — SDA Output Delay (Tdel) Selection (See Figure 22-12)

00: 300ns

*Table 22-4. I2C Command Register (I2CCMDR)*

| I2CCMDR | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | STA | STO | RD | WR | ACK | CKSDIS | RBUFDIS | (Reserved) |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 to disable | YES | YES | YES | - | - | No | No | - |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | - |

STA — Generate START (or Repeated START) condition (Master operation)

STO — Generate STOP condition (Master operation)

RD — Indicate Read from slave (Master operation)

WR — Indicate Write to slave (Master operation)

ACK — Acknowledge Option – when receiving, ACK transmission selection

0: Send ACK
1: Send NACK

CKSDIS — Clock Stretching Disable – Disables the clock stretching if desired by the user for both master and slave mode. Then overflow error flag must be monitored.0:Send ACK

0: Enable Clock Stretching
1: Disable Clock Stretching

RBUFDIS — Read Command with Buffer Disable – Read from Slave in master mode with the double buffering disabled for easier control over single byte data communication scenario.

0: Read with buffer enabled as default
1: Read with buffer disabled

***Table 22-5. I2C Clock Pre-scale Register (I2CCBR)***

| I2CCBR | MSB[1:0] | | LSB[7:0] | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Bit | Bit9 | Bit8 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | I2C_PRESCALE | | | | | | | | | |
| Default | 0000000000 | | | | | | | | | |
| Access | R/W | | | | | | | | | |

**I2C_PRESCALE[9:0]**
I2C Clock Pre-scale value. A write operation to I2CBR [9:8] will cause an I2C core reset. The System Bus clock frequency is divided by (I2C_PRESCALE*4) to produce the Master I2C clock frequency supported by the I2C bus (50KHz, 100KHz, 400KHz).

***Table 22-6. I2C Status Register (I2CCSR)***

| I2CCSR | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | TIP | BUSY | RARC | SRW | ARBL | TRRDY | TROE | HGC |
| Default | - | - | - | - | - | - | - | - |
| Access | R | R | R | R | R | R | R | R |

TIP — Transmitting In Progress - This bit indicates that current data byte is being transferred for both master and slave mode Note that the TIP flag will suffer half SCL cycle latency right after the start condition because of the signal synchronization. Also note that this bit could be high after configuration wake-up and before the first valid I2C transfer start (when BUSY is low), and it is not indicating byte in transfer, but an invalid indicator.

0: Byte transfer completed
1: Byte transfer in progress

BUSY — Bus Busy – This bit indicates the bus is involved in transaction. This will be set at start condition and cleared at stop. Therefore, only when this bit is high, should all other status bits be treated as valid indicators for a valid transfer.

RARC — Received Acknowledge – This flag represents acknowledge response from the addressed slave during master write or from receiving master during master read.

0: No acknowledge received
1: Acknowledge received

SRW — Slave RW

0: Master transmitting / Slave receiving
1: Master receiving / Slave transmitting

ARBL — Arbitration Lost – This bit will go high if master has lost its arbitration in Master mode, It will cause an interrupt to System Bus Host if SCI set up allowed.

0: Normal
1: Arbitration Lost

TRRDY — Transmitter or Receiver Ready Bit – This flag indicate that a Transmit Register ready to receive data or Receiver Register if ready for read depend on the mode (master or slave) and SRW bit. It will cause an interrupt to System Bus Host if SCI set up allowed.

0: Transmitter or Receiver is not ready
1: Transmitter or Receiver is ready

TROE — Transmitter/Receiver Overrun or NACK Received Bit – This flag indicate that a Transmit or Receive Overrun Errors happened depend on the mode (master or slave) and SRW bit, or a no-acknowledges response is received after transmitting a byte. If RARC bit is high, it is a NACK bit, otherwise, it is overrun bit. It will cause an interrupt to System Bus Host if SCI set up allowed.

0: Transmitter or Receiver Normal or Acknowledge Received for Transmitting
1: Transmitter or Receiver Overrun or No-Acknowledge Received for Transmitting

HGC — Hardware General Call Received – This flag indicate that a hardware general call is received from the slave port. It will cause an interrupt to System Bus Host if SCI set up allowed.

0: NO Hardware General Call Received in Slave Mode
1: Hardware General Call Received in Slave Mode

***Table 22-7. I2C Transmitting Data Register (I2CTXDR)***

| I2CTXDR | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | I2C_Transmit_Data[7:0] | | | | | | | |
| Default | 00000000 | | | | | | | |
| Access | W | | | | | | | |

I2C_ Transmit_Data[7:0] — I2C Transmit Data – This register holds the byte that will be transmitted on the I2C bus during the Write Data phase. Bit 0 is the LSB and will be transmitted last. When transmitting the slave address, Bit 0 represents the Read/Write bit.

***Table 22-8. I2C Receiving Data Register (I2CRXDR)***

| I2CTXDR | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | I2C_Receive_Data[7:0] | | | | | | | |
| Default | - | | | | | | | |
| Access | R | | | | | | | |

I2C_ Receive _Data[7:0] I2C Receive Data – This register holds the byte captured from the I2C bus during the Read Data phase. Bit 0 is LSB and was received last.

***Table 22-9. I2C General Call Data Register (I2CGCDR)***

| I2CGCDR | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | I2C_GC_Data[7:0] | | | | | | | |
| Default | - | | | | | | | |
| Access | R | | | | | | | |

I2C_ GC _Data[7:0] I2C General Call Data – This register holds the second (command) byte of the General Call transaction on the I2C bus.

***Table 22-10. I2C Slave Address MSB Register (I2CSADDR)***

| I2CSADDR | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| 7 Bits Addressing | - | - | - | A6 | A5 | A4 | A3 | A2 |
| 10 Bits Addressing | A9 | A8 | A7 | A6 | A5 | A4 | A3 | A2 |
| Default | 00000000 | | | | | | | |
| Access | R/W | | | | | | | |

***Table 22-11. I2C Interrupt Control Register (I2CIRQEN)***

| I2CIRQEN | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | IRQINTCLREN | IRQINTFRC | RSVD | RSVD | IRQARBLEN | IRQTRRDYEN | IRQTROEEN | IRQHGCEN |
| Default | 0 | 0 | - | - | 0 | 0 | 0 | 0 |
| 0 to Disable | YES | YES | - | - | YES | Yes | YES | YES |
| Access | R/W | R/W | - | - | R/W | R/W | R/W | R/W |

IRQINTCLREN Auto Interrupt Clear Enable – Enable the interrupt flag auto clear when the I2CIRQ has been read.

IRQINTFRC Force Interrupt Request On – Force the Interrupt Flag set to improve testability

IRQARBLEN Interrupt Enable for Arbitration Lost

IRQTRRDYEN Interrupt Enable for Transmitter or Receiver Ready

IRQTROEEN Interrupt Enable for Transmitter/Receiver Overrun or NACK Received

IRQHGCEN Interrupt Enable for Hardware General Call Received

***Table 22-12. I2C Interrupt Status Register (I2CIRQ)***

| I2CIRQ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Bit** | **Bit7** | **Bit6** | **Bit5** | **Bit4** | **Bit3** | **Bit2** | **Bit1** | **Bit0** |
| Name | (Reserved) | | | | IRQARBL | IRQTRRDY | IRQTROE | IRQHGC |
| Default | - | - | - | - | - | - | - | - |
| Access | - | - | - | - | R/W | R/W | R/W | R/W |

| | |
|---|---|
| IRQARBL | Interrupt Status for Arbitration Lost.<br>When enabled, indicates ARBL was asserted. Write a '1' to this bit to clear the interrupt.<br>0: No interrupt<br>1: Arbitration Lost Interrupt |
| IRQTRRDY | Interrupt Status for Transmitter or Receiver Ready.<br>When enabled, indicates TRRDY was asserted. Write a '1' to this bit to clear the interrupt.<br>0: No interrupt<br>1: Transmitter or Receiver Ready Interrupt |
| IRQTROE | Interrupt Status for Transmitter/Receiver Overrun or NACK received.<br>When enabled, indicates TROE was asserted. Write a '1' to this bit to clear the interrupt.<br>0: No interrupt<br>1: Transmitter or Receiver Overrun or NACK received Interrupt |
| IRQHGC | Interrupt Status for Hardware General Call Received.<br>When enabled, indicates HGC was asserted. Write a '1' to this bit to clear the interrupt.<br>0: No interrupt<br>1: General Call Received in slave mode Interrupt |

## I2C Read/Write Flow Chart

Figure 22-5 shows a flow diagram for controlling Master I2C reads and writes initiated via the System Bus interface.

***Figure 22-5. I2C Master Read/Write Example (via System Bus)***

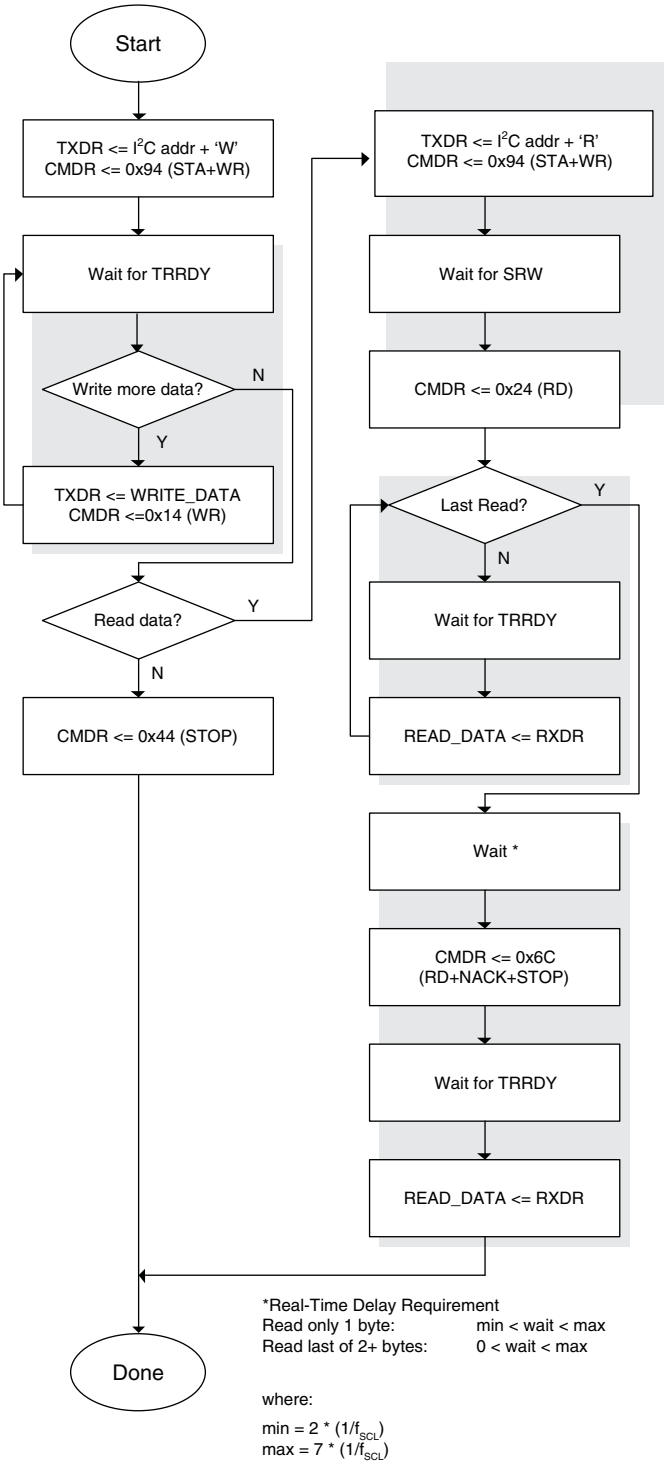
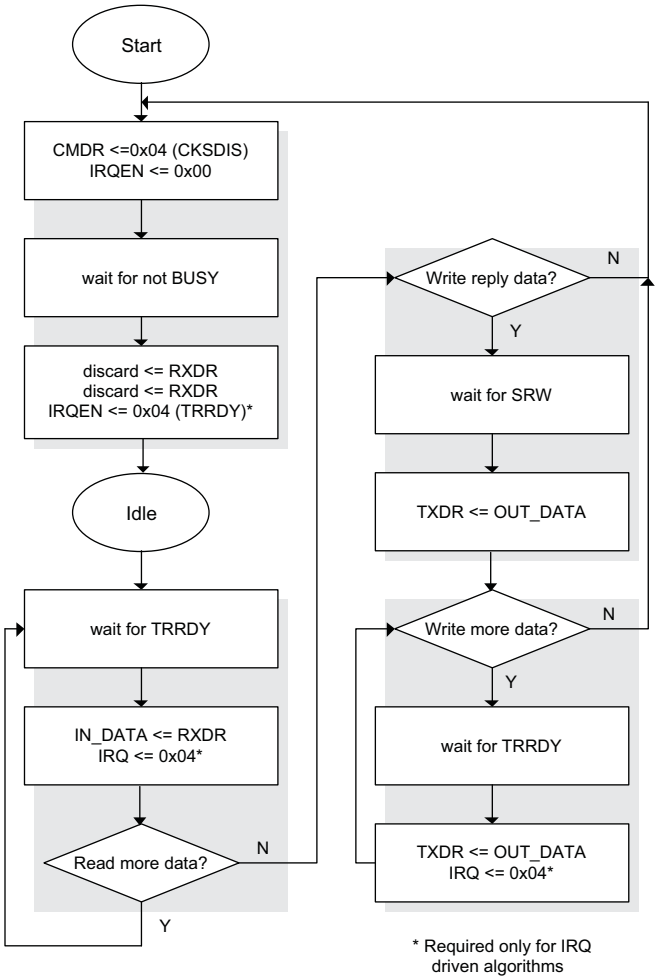


Figure 22-6 shows a flow diagram for reading and writing from an I2C Slave device via the System Bus interface.

***Figure 22-6. I2C Slave Read/Write Example (via System Bus)***



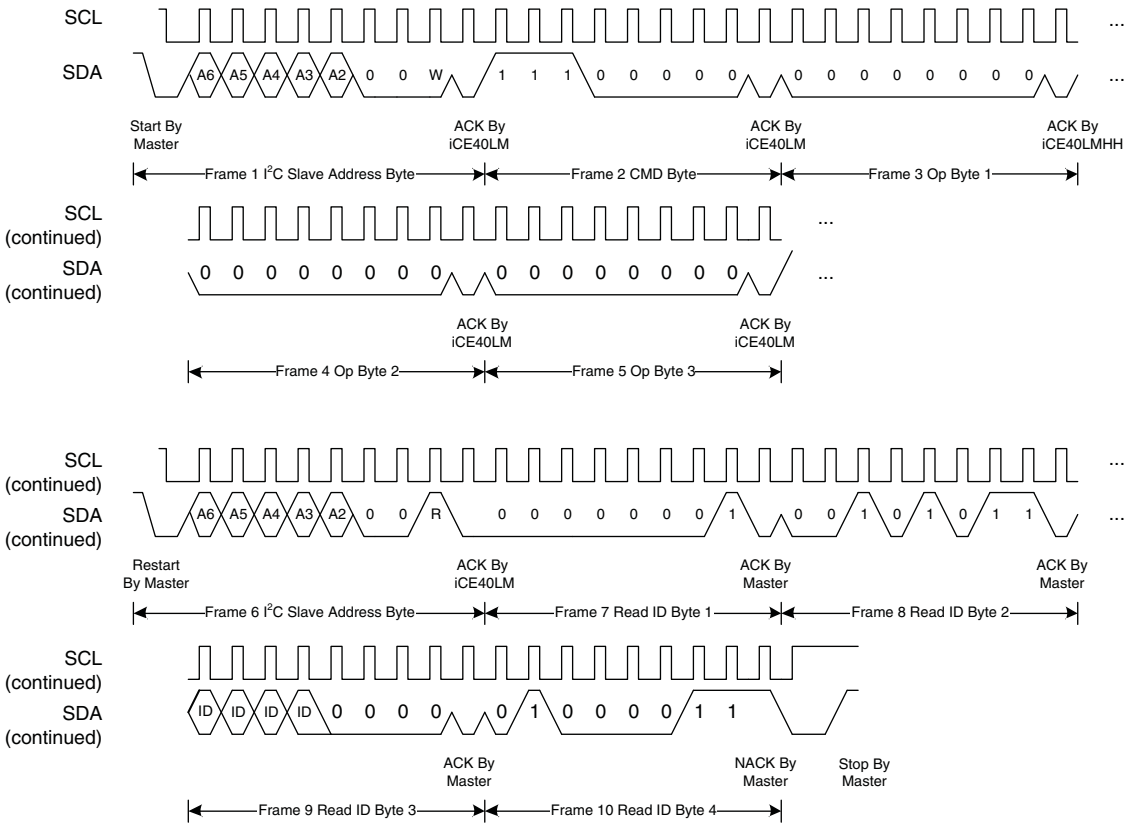
## I2C Framing

Each command string sent to the I2C port must be correctly "framed" using the protocol defined for each interface. In the case of I2C, the protocol is well known and defined by the industry as shown below.

***Table 22-13. Command Framing Protocol, by Interface***

| Interface | Pre-op (+) | Command String | Post-op (-) |
|---|---|---|---|
| I2C | Start | (Command/Operands/Data) | Stop |

***Figure 22-7. I2C Read Device ID Example***



# I2C Functional Waveforms

***Figure 22-8. Master – I2C Write***

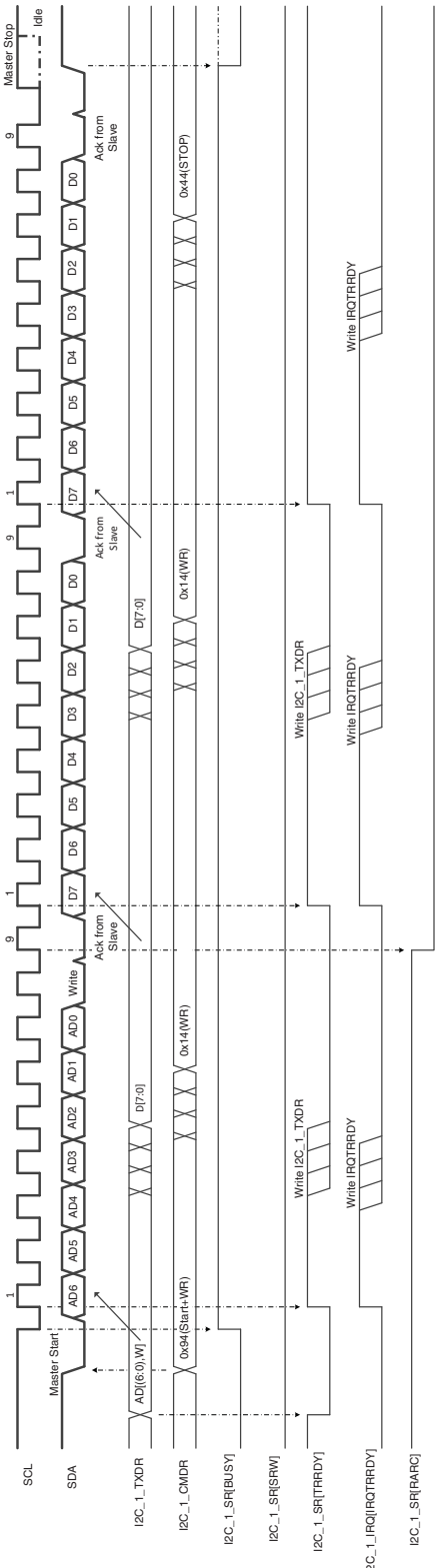



***Figure 22-9. Master – I2C Read***

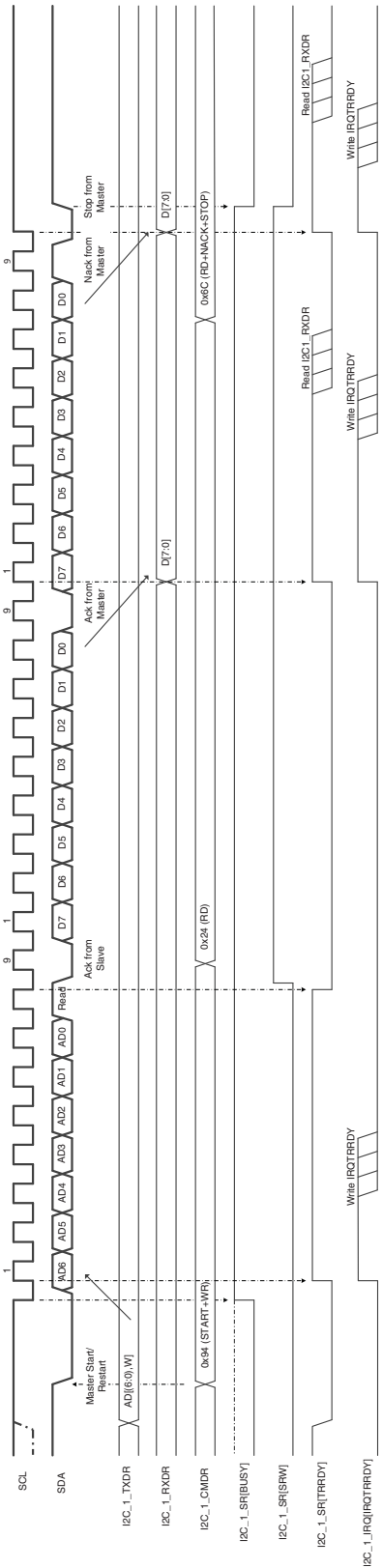


*Figure 22-10. Slave – I2C Write*

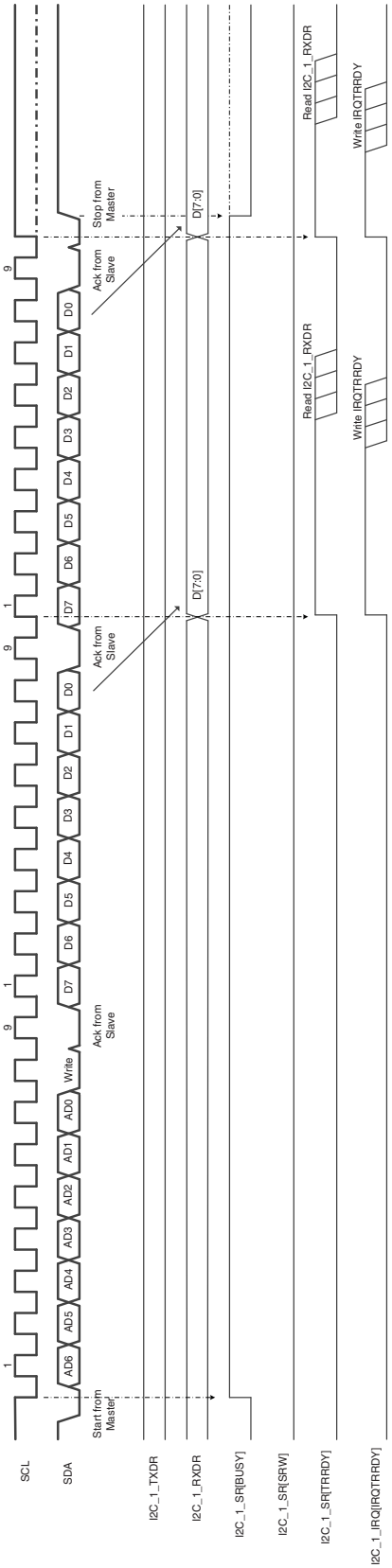
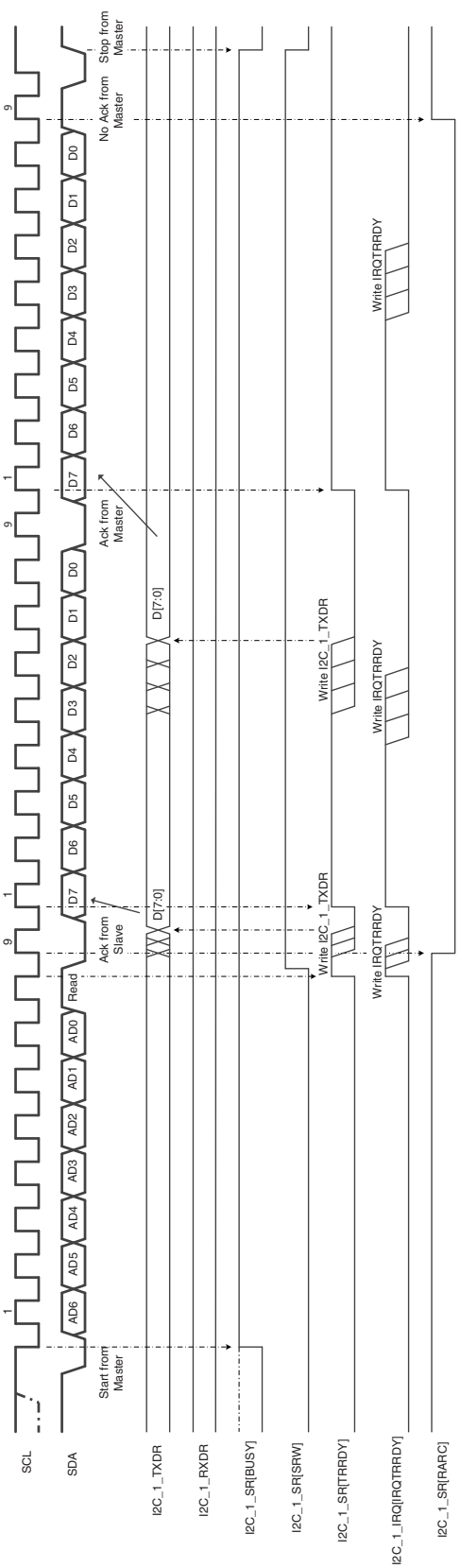
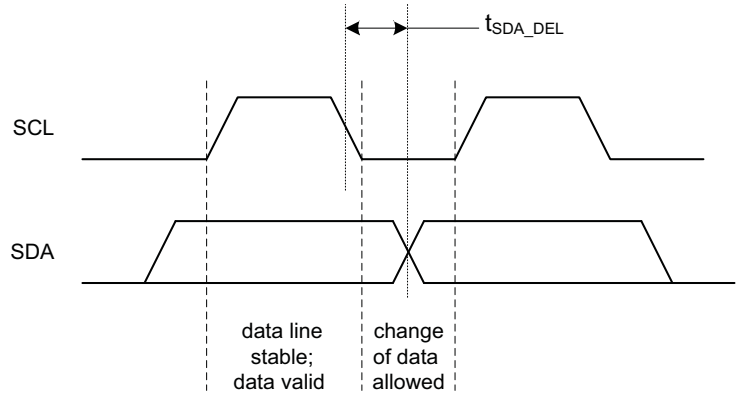



*Figure 22-11. Slave – I2C Read*



## I2C Timing Diagram

***Figure 22-12. I2C Bit Transfer Timing***




## Hardened SPI IP Core

The iCE40LM contains two hard SPI IP cores that can be configured as a SPI Master or Slave. When the SPI core is configured as a Master it is able to control other devices with Slave SPI interfaces that are connected to the SPI bus. When the SPI core is configured as a Slave, it is able to interface to an external SPI Master device.

The SPI core communicates with the System Bus interface through a set of control, command, status and data registers. Table 2 shows the register names and their functions.

## SPI Registers

***Table 22-14. SPI Registers Summary***

| SPI Register Name | Simulation Model Register Name | Address[3:0] | Register Function | Access |
|---|---|---|---|---|
| SPICR0 | SPICR0 | 1000 | SPI Control Register 0 | Read/Write |
| SPICR1 | SPICR1 | 1001 | SPI Control Register 1 | Read/Write |
| SPICR2 | SPICR2 | 1010 | SPI Control Register 2 | Read/Write |
| SPIBR | SPIBR | 1011 | SPI Baud Rate Register | Read/Write |
| SPISR | SPISR | 1100 | SPI Status Register | Read |
| SPITXDR | SPITXDR | 1101 | SPI Transmit Data Register | Read/Write |
| SPIRXDR | SPIRXDR | 1110 | SPI Receive Data Register | Read |
| SPICSR | SPICSR | 1111 | SPI Chip Select Mask For Master Mode | Read/Write |
| SPIIRQEN | SPIINTCR | 0111 | SPI Interrupt Control Register | Read/Write |
| SPIIRQ | SPIINTSR | 0110 | SPI Interrupt Status Register | Read/Write[1] |

1. SPIIRQ is Read Only. Write operation upon this register will not change the content of this register, but will clear corresponding interrupt flag caused by the flags inside SPIIRQ.

*Table 22-15. SPI Control Register 0 (SPICR0)[1]*

| SPICR0 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | TIdle_XCNT[1:0] | | TTrail_XCNT[2:0] | | | TLead_XCNT[2:0] | | |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | - | R/W | - | - | |

1. A write to this register will cause the SPI core to reset

TIdle_XCNT[1:0] — Idle Delay Count – Specifies the minimum interval prior to the Master Chip Select low assertion (Master Mode only), in SCK periods.

00: ½
01: 1
10: 1.5
11: 2

TTrail_XCNT[2:0] — Trail Delay Count – Specifies the minimum interval between the last edge of SCK and the high deassertion of Master Chip Select (Master Mode only), in SCK periods.

000: ½
001: 1
010: 1.5
…
111: 4

TLead_XCNT[2:0] — Lead Delay Count – Specifies the minimum interval between the Master Chip Select low assertion and the first edge of SCK (Master Mode only), in SCK periods.

000: ½
001: 1
010: 1.5
…
111: 4

*Table 22-16. SPI Control Register 1 (SPICR1)[1]*

| SPICR1 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | SPE | WKUPEN_USER | (Reserved) | TXEDGE | (Reserved) | | | |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | - | R/W | - | - | - | - |

1. A write to this register will cause the SPI core to reset

SPE — This bit enables the SPI core functions. If SPE is cleared, SPI is disabled and forced into idle state.

0: SPI disabled
1: SPI enabled, port pins are dedicated to SPI functions.

WKUPEN_USER — Wake-up Enable via User – Enables the SPI core to send a wake-up signal to the on-chip Power Controller to wake the part from Standby mode when the User slave SPI chip select (spi_scsn) is driven low.

0: Wakeup disabled
1: Wakeup enabled.

WKUPEN_CFG — Wake-up Enable Configuration – Enables the SPI core to send a wake-up signal to the on-chip power controller to wake the part from standby mode when the Configuration slave SPI chip select (ufm_sn) is driven low.

0: Wakeup disabled
1: Wakeup enabled.

TXEDGE — Data Transmitting selection bit – This bit gives user capability to select which clock edge to transmit data for fast SPI applications. Note that this bit should not be set when CPHA or MCSH of SPICR2 is set.

0: Transmit data on the different clock edge of data receiving (receiving on rising / transmit on falling)
1: Transmit data on the same clock edge of data receiving (receiving on rising /transmit on rising)

***Table 22-17. SPI Control Register 2 (SPICR2) [1]***

| SPICR2 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | MSTR | MCSH | SDBRE | (Reserved) | | CPOL | CPHA | LSBF |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | - | - | R/W | R/W | R/W |

1. A write to this register will cause the SPI core to reset

MSTR — SPI Master/Slave Mode – Selects the Master/Slave operation mode of the SPI core. Changing this bit forces the SPI system into idle state.

0: SPI is in Slave mode
1: SPI is in Master mode

MCSH — SPI Master CSSPIN Hold – Holds the Master chip select active when the host is busy, to halt the data transmission without de-asserting chip select.

*Note: This mode must be used only when the System Bus clock has been divided by a value greater than three (3).*

0: Master running as normal
1: Master holds chip select low even if there is no data to be transmitted

SDBRE — Slave Dummy Byte Response Enable – Enables Lattice proprietary extension to the SPI protocol. For use when the internal support circuit (e.g. System host) cannot respond with initial data within the time required, and to make the slave read out data predictably available at high SPI clock rates.

When enabled, dummy 0xFF bytes will be transmitted in response to a SPI slave read (while SPISR[TRDY]=1) until an initial write to SPITXDR. Once a byte is written into SPITXDR by the System host, a single byte of 0x00 will be transmitted then fol-lowed immediately by the data in SPITXDR. In this mode, the external SPI master should scan for the initial 0x00 byte when reading the SPI slave to indicate the begin-ning of actual data. Refer to Figure 22-16

0: Normal Slave SPI operation
1: Lattice proprietary Slave Dummy Byte Response Enabled

*Note: This mechanism only applies for the initial data delay period. Once the initial data is available, subsequent data must be supplied to SPITXDR at the required SPI bus data rate.*

CPOL — SPI Clock Polarity – Selects an inverted or non-inverted SPI clock. To transmit data between SPI modules, the SPI modules must have identical SPICR2[CPOL] values. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. Refer to Figures XX through XX.

0: Active-high clocks selected. In idle state SCK is low.
1: Active-low clocks selected. In idle state SCK is high.

CPHA — SPI Clock Phase – Selects the SPI clock format. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. Refer to Refer to Figures XX through XX.

0: Data is captured on a leading (first) clock edge, and propagated on the opposite clock edge.
1: Data is captured on a trailing (second) clock edge, and propagated on the opposite clock edge*.

*Note: When CPHA=1, the user must explicitly place a pull-up or pull-down on SCK pad corresponding to the value of CPOL (e.g. when CPHA=1 and CPOL=0 place a pull-down on SCK). When CPHA=0, the pull direction may be set arbitrarily.*

Slave SPI Configuration mode supports default setting only for CPOL, CPHA.

LSBF — LSB-First – LSB appears first on the SPI interface. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. Refer to Figures XX through XX.

*Note: This bit does not affect the position of the MSB and LSB in the data register. Reads and writes of the data register always have the MSB in bit 7.*

0: Data is transferred most significant bit (MSB) first
1: Data is transferred least significant bit (LSB) first

***Table 22-18. SPI Clock Prescale (SPIBR)***

| SPIBR | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | (Reserved) | | DIVIDER[5:0] | | | | | |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | - | - | R/W | R/W | R/W | R/W | R/W | R/W |

DIVIDER[5:0] — SPI Clock Prescale value – The System clock frequency is divided by (DIVIDER[5:0] + 1) to produce the desired SPI clock frequency. A write operation to this register will cause a SPI core reset. DIVIDER must be >= 1.

*Note: The digital value is calculated by Module Generator when the SPI core is configured in the SPI tab of the Module Generator GUI. The calculation is based on the System Bus Clock Frequency and the SPI Frequency, both entered by the user. The digital value of the divider is loaded in the iCE40LM device using Soft IP into the SPIBR register.*

Register SPIBR has Read/Write access from the System Bus interface. Designers can update the clock pre-scale register dynamically during device operation.

*Table 22-19. SPI Master Chip Select Register (SPICSR)*

| SPICSR | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | (Reserved) | | | | CSN_3 | CSN_2 | CSN_1 | CSN_0 |
| Default | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | - | - | - | - | R/W | R/W | R/W | R/W |

CSN_[7:0] SPI Master Chip Selects – Used in master mode for asserting a specific Master Chip Select (MCSN) line. The register has four bits, enabling the SPI core to control up to four external SPI slave devices. Each bit represents one master chip select line (Active-Low). Bits [3:1] may be connected to any I/O pin via the FPGA fabric. Bit 0 has a pre-assigned pin location. The register has Read/Write access from the System Bus interface. A write operation on this register will cause the SPI core to reset.

*Table 22-20. SPI Status Register(SPISR)*

| SPISR | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | TIP | BUSY | (Reserved) | TRDY | RRDY | TOE | ROE | MDF |
| Default | - | - | - | - | - | - | - | 0 |
| Access | R | R | - | R | R | R | R | R |

TIP SPI Transmitting In Progress – Indicates the SPI port is actively transmitting/receiving data.
- 0: SPI Transmitting complete
- 1: SPI Transmitting in progress*

BUSY SPI Busy Flag – This bit indicate that the SPI port in the middle of data transmitting / receiving (CSN is low)
- 0: SPI Transmitting complete
- 1: SPI Transmitting in progress*

TRDY SPI Transmit Ready – Indicates the SPI transmit data register (SPITXDR) is empty. This bit is cleared by a write to SPITXDR. This bit is capable of generating an interrupt.
- 0: SPITXDR is not empty
- 1: SPITXDR is empty

RRDY SPI Receive Ready – Indicates the receive data register (SPIRXDR) contains valid receive data. This bit is cleared by a read access to SPIRXDR. This bit is capable of generating an interrupt.
- 0: SPIRXDR does not contain data
- 1: SPIRXDR contains valid receive data

TOE Receive Overrun Error – This bit indicates that the SPIRXDR received new data before the previous data was read. The previous data will be lost if occurs. It will cause an interrupt to System Host if SCI set up allowed.
- 0: Normal
- 1: Transmit Overrun detected

ROE Receive Overrun Error – Indicates SPIRXDR received new data before the previous data was read. The previous data is lost. This bit is capable of generating an interrupt.
- 0: Normal
- 1: Receiver Overrun detected

MDF — Mode Fault – Indicates the Slave SPI chip select (spi_scsn) was driven low while SPICR2[MSTR]=1. This bit is cleared by any write to SPICR0, SPICR1 or SPICR2. This bit is capable of generating an interrupt.

0: Normal
1: Mode Fault detected

***Table 22-21. SPI Transmit Data Register (SPITXDR)***

| SPITXDR | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | SPI_Transmit_Data[7:0] | | | | | | | |
| Default | - | - | - | - | - | - | - | - |
| Access | W | W | W | W | W | W | W | W |

SPI_Receive_Data[7:0] — SPI Transmit Data – This register holds the byte that will be transmitted on the SPI bus. Bit 0 in this register is LSB, and will be transmitted last when SPICR2[LSBF]=0 or first when SPICR2[LSBF]=1.

*Note: When operating as a Slave, SPITXDR must be written when SPISR[TRDY] is '1' and at least 0.5 CCLKs before the first bit is to appear on SO. For example, when CPOL = CPHA = TXEDGE = LSBF = 0, SPITXDR must be written prior to the CCLK rising edge used to sample the LSB (bit 0) of the previous byte. See Figure 17-25. This timing requires at least one protocol dummy byte be included for all slave SPI read operations.*

***Table 22-22. SPI Receive Data Register (SPIRXDR)***

| SPIRXDR | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | SPI_Receive_Data[7:0] | | | | | | | |
| Default | - | - | - | - | - | - | - | - |
| Access | R | R | R | R | R | R | R | R |

SPI_Receive_Data[7:0] — SPI Receive Data This register holds the byte captured from the SPI bus. Bit 0 in this register is LSB and was received last when LSBF=0 or first when LSBF=1.

***Table 22-23. SPI Interrupt Status Register (SPIIRQ)***

| SPIIRQ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | (Reserved) | | | IRQTRDY | IRQRRDY | IRQTOE | IRQROE | IRQMDF |
| Default | - | - | - | - | - | - | - | - |
| 0 means No Interrupt | - | - | - | YES | YES | YES | YES | YEs |
| Access | - | - | - | R/W | R/W | R/W | R/W | R/W |

IRQTRDY — Interrupt Status for SPI Transmit Ready.
When enabled, indicates SPISR[TRDY] was asserted.
Write a ‘1’ to this bit to clear the interrupt.

IRQRRDY — Interrupt Status for SPI Receive Ready.
When enabled, indicates SPISR[RRDY] was asserted.
Write a ‘1’ to this bit to clear the interrupt.

IRQROE — Interrupt Status for Receive Overrun Error.
When enabled, indicates ROE was asserted.
Write a '1' to this bit to clear the interrupt.

IRQMDF — Interrupt Status for Mode Fault.
When enabled, indicates MDF was asserted.
Write a '1' to this bit to clear the interrupt.

***Table 22-24. SPI Interrupt Enable Register (SPIIRQEN)***

| SPIIRQ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Bit | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
| Name | (Reserved) | | | IRQTRDYEN | IRQRRDYEN | IRQTOEEN | IRQROEEN | IRQMDFEN |
| Default | - | - | - | - | - | - | - | - |
| 0 means Disable | - | - | - | YES | YES | YES | YES | YEs |
| Access | - | - | - | R/W | R/W | R/W | R/W | R/W |

IRQTRDYEN — Interrupt Enable for SPI Transmit Ready.

IRQRRDYEN — Interrupt Enable for SPI Receive Ready

IRQTOEEN — Interrupt Enable for SPI Transmit Overrun Ready.

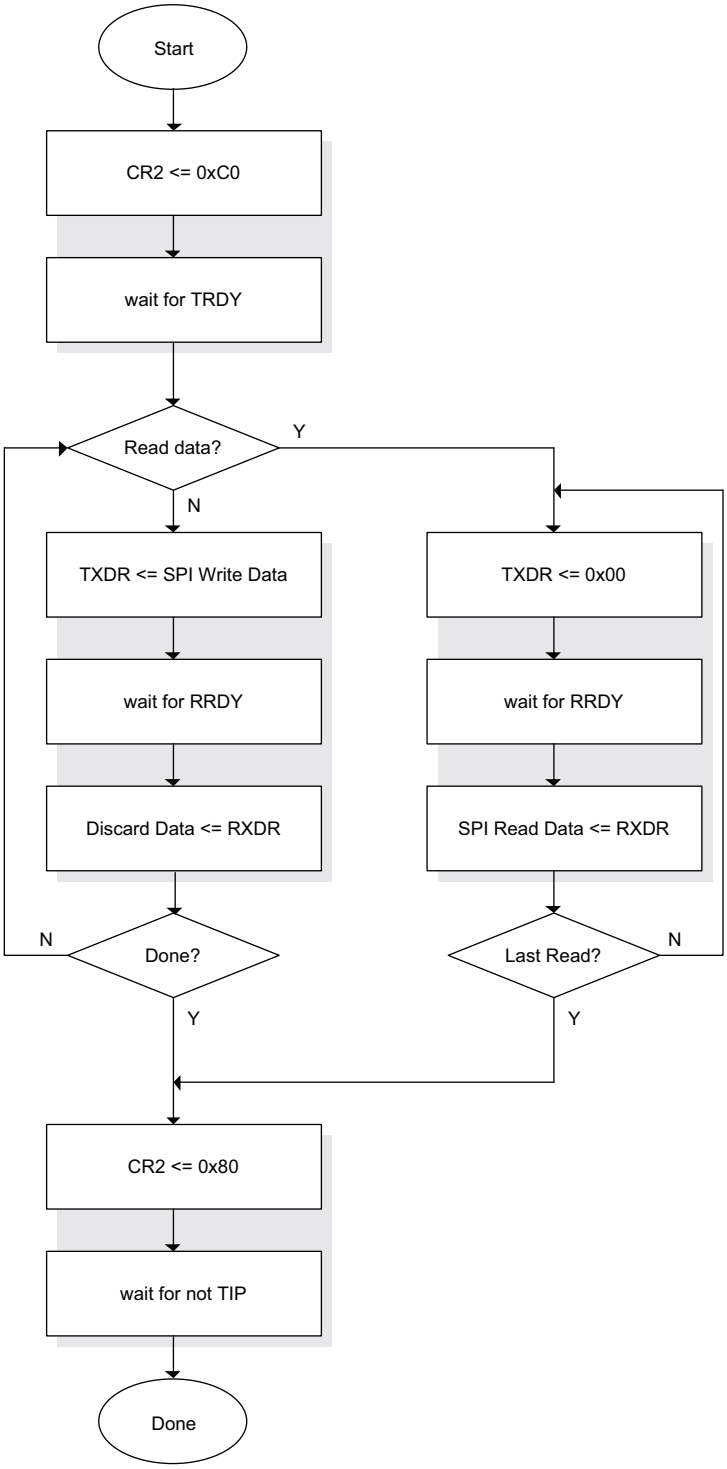
IRQROEEN — Interrupt Enable for SPI Receive Overrun Ready.

IRQMDFEN — Interrupt Enable for SPI Mode Default Ready.

## SPI Read/Write Flow Chart

Figure 22-13 shows a flow diagram for controlling Master SPI reads and writes initiated via the System Bus interface.

***Figure 22-13. SPI Master Read/Write Example (via System Bus)***



*Note: Assumes CR2 register, MSCH = '1'. The algorithm when MSCH = '0' is application dependent and not provided. See Figure 22-15 for guidance.*

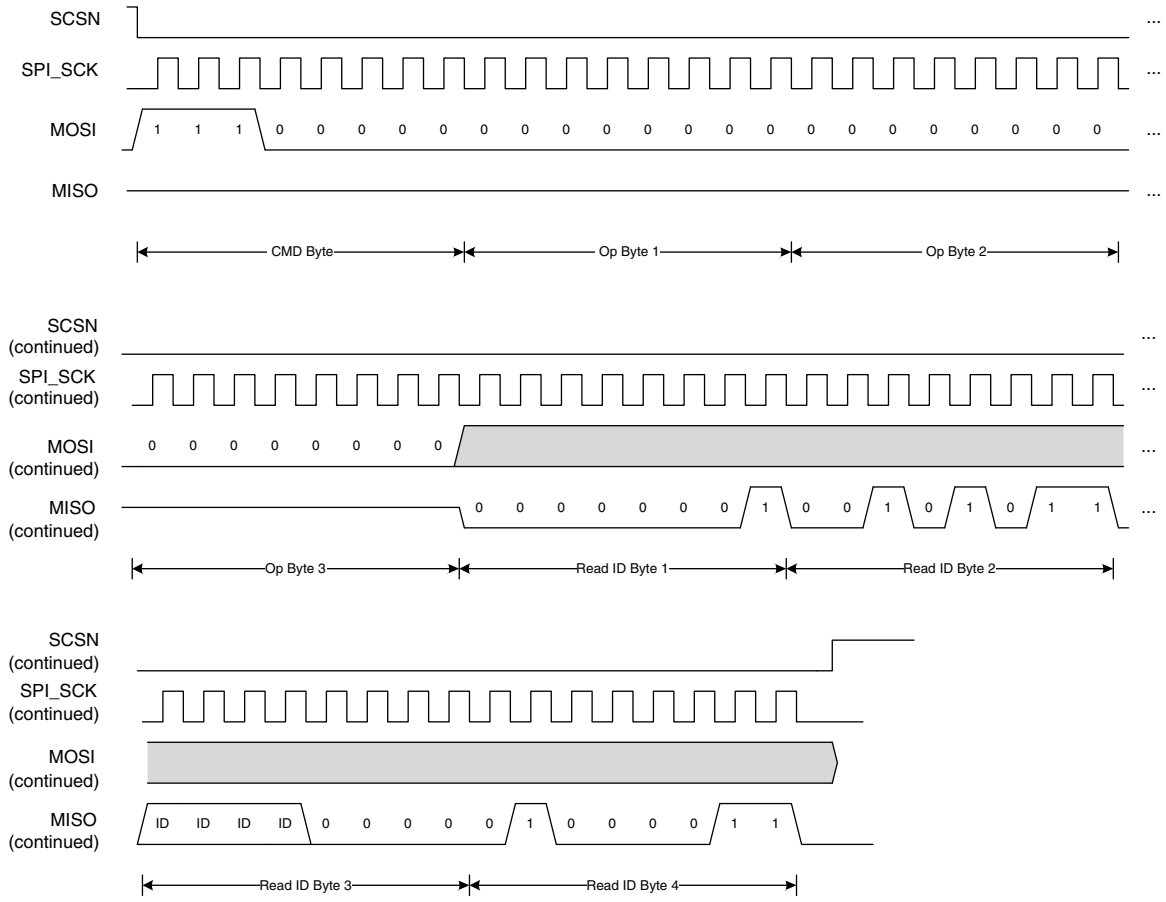
## SPI Framing

Each command string sent to the SPI port must be correctly 'framed' using the protocol defined for each interface. In the case of SSPI the protocol is well known and defined by the industry as shown below:

***Table 22-25. Command Framing Protocol, by Interface***

| Interface | Pre-op (+) | Command String | Post-op (-) |
|---|---|---|---|
| SPI | Assert CS | (Command/Operands/Data) | De-assert CS |

***Figure 22-14. SSPI Read Device ID Example***



# SPI Functional Waveforms

***Figure 22-15. Fully Specified SPI Transaction (iCE40LM as SPI Master or Slave)***

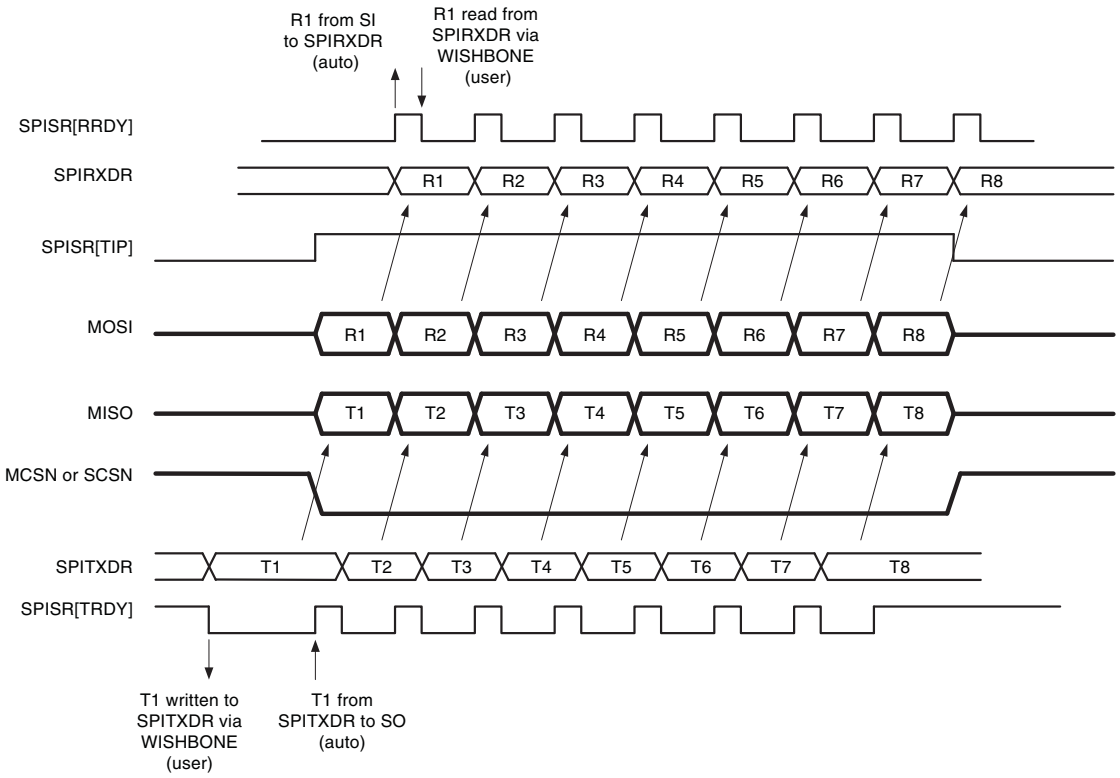
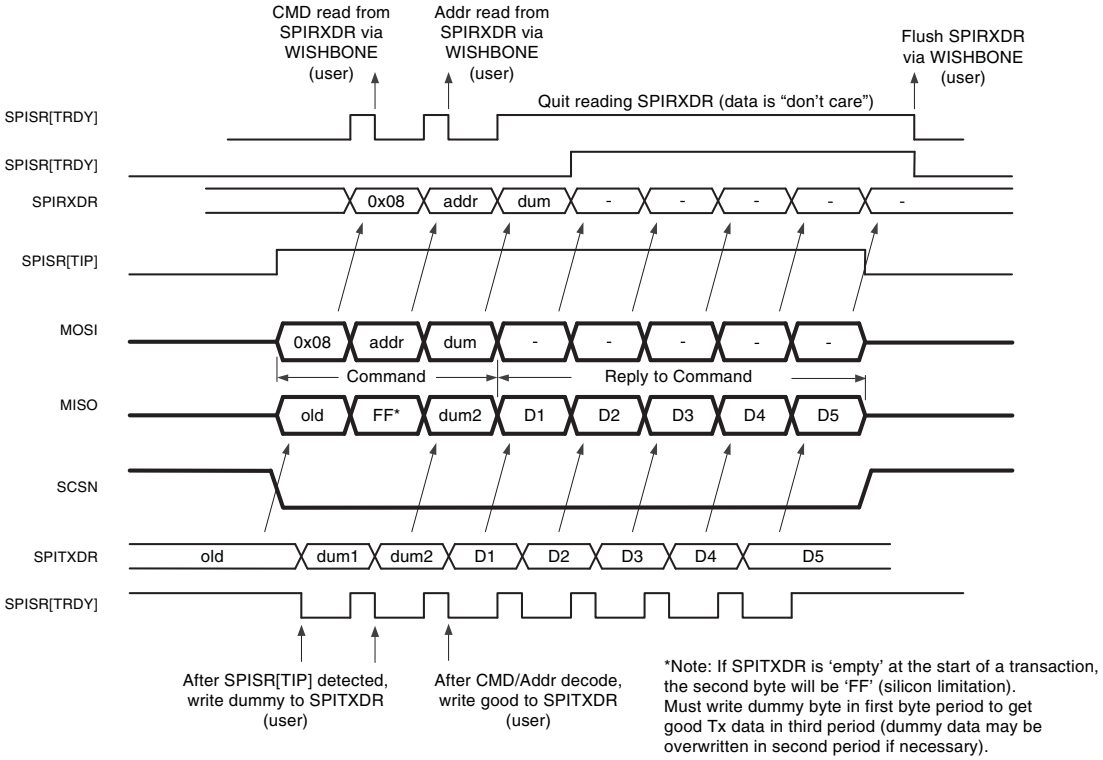
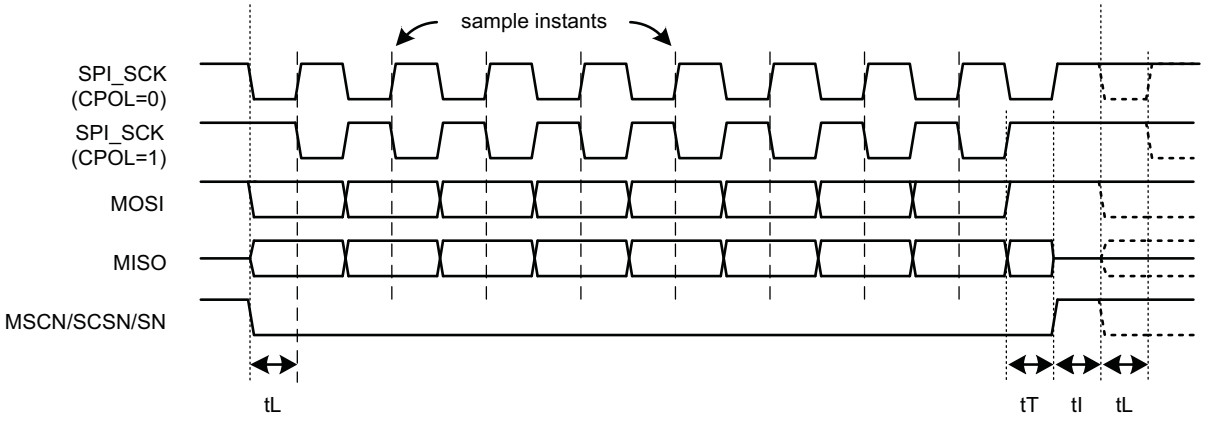



***Figure 22-16. Minimally Specified SPI Transaction Example (iCE40LM as SPI Slave)***



## SPI Timing Diagrams

***Figure 22-17. SPI Control Timing (SPICR2[CPHA]=0, SPICR1[TXEDGE]=0)***

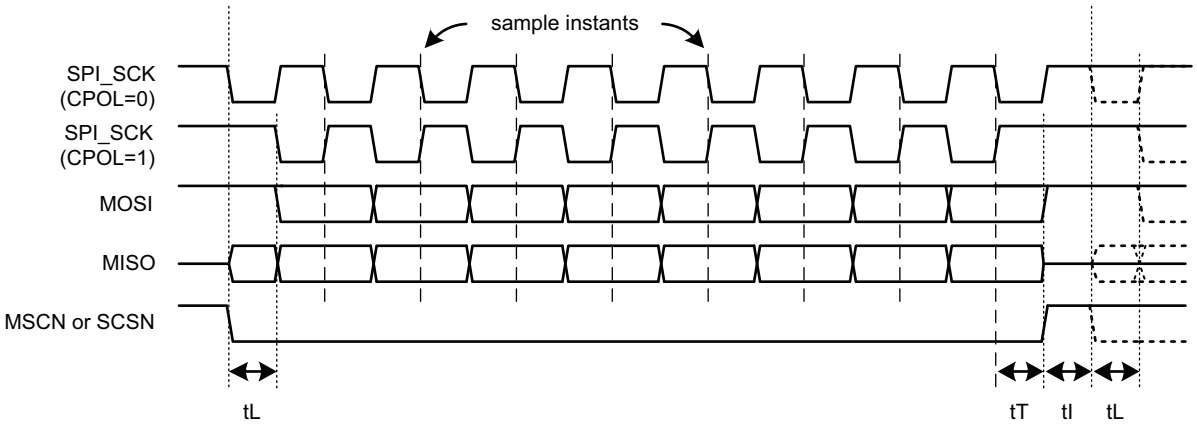



| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| MSB first (LSBF=0): | MSB | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | LSB |
| LSB first (LSBF=1): | LSB | bit1 | bit2 | bit3 | bit4 | bit5 | bit6 | MSB |

tL = TLead_XCNT
tT = TTrail_XCNT
tL = Tidle_XCNT

*Note: iCE40LM supports only
CPHA = CPOL = LSBF = TXEDGE = 0

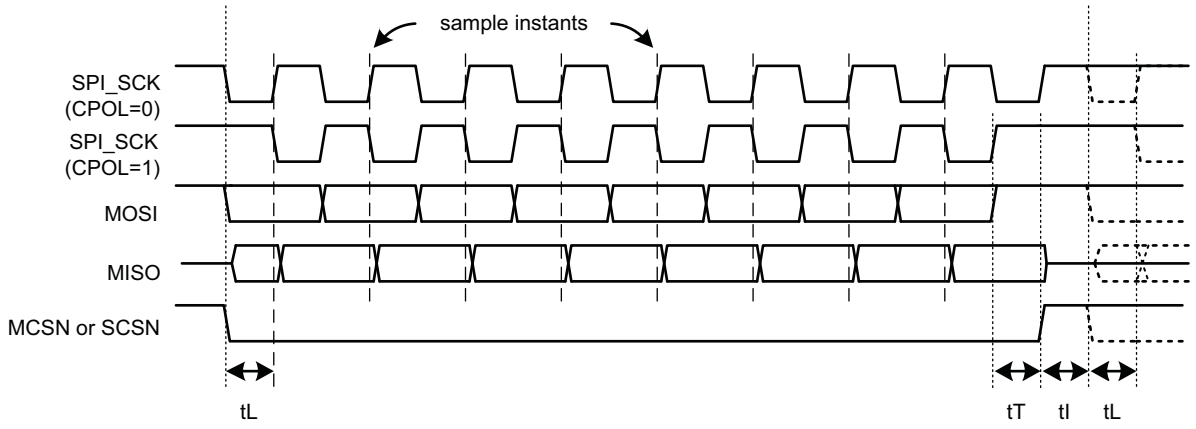
***Figure 22-18. SPI Control Timing (SPICR2[CPHA]=1, SPICR1[TXEDGE]=0)***




| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| MSB first (LSBF=0): | MSB | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | LSB |
| LSB first (LSBF=1): | LSB | bit1 | bit2 | bit3 | bit4 | bit5 | bit6 | MSB |

tL = TLead_XCNT
tT = TTrail_XCNT
tL = Tidle_XCNT

*Figure 22-19. SPI Control Timing (SPICR2[CPHA]=0, SPICR1[TXEDGE]=1)*




| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| MSB first (LSBF=0): | MSB | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | LSB |
| LSB first (LSBF=1): | LSB | bit1 | bit2 | bit3 | bit4 | bit5 | bit6 | MSB |

tL = TLead_XCNT

tT = TTrail_XCNT

tL = Tidle_XCNT

## Technical Support Assistance

e-mail: techsupport@latticesemi.com

Internet: www.latticesemi.com

## Revision History

| Date | Version | Change Summary |
|---|---|---|
| October 2013 | 01.0 | Initial release. |

# Section IV. iCE40 LP/HX/LM Family Handbook Revision History

# iCE40 LP/HX/LM Family Handbook Revision History



## Revision History

| Date | Handbook Revision Number | Change Summary |
|---|---|---|
| October 2012 | 01.0 | Initial release. |
| October 2013 | 01.1 | Changed document name to ice40 LP/HX/LM Family Handbook |
| November 2013 | 01.2 | Added the DC and Switching Characteristics, Pinout Information, and Further Information chapters in Sections I and II. |
| | | Technical note TN1251 updated to version 01.3. |

Note: For detailed revision changes, please refer to the revision history for each document.