



## iCEblink40-LP1K Evaluation Kit

---

## User's Guide

## Introduction

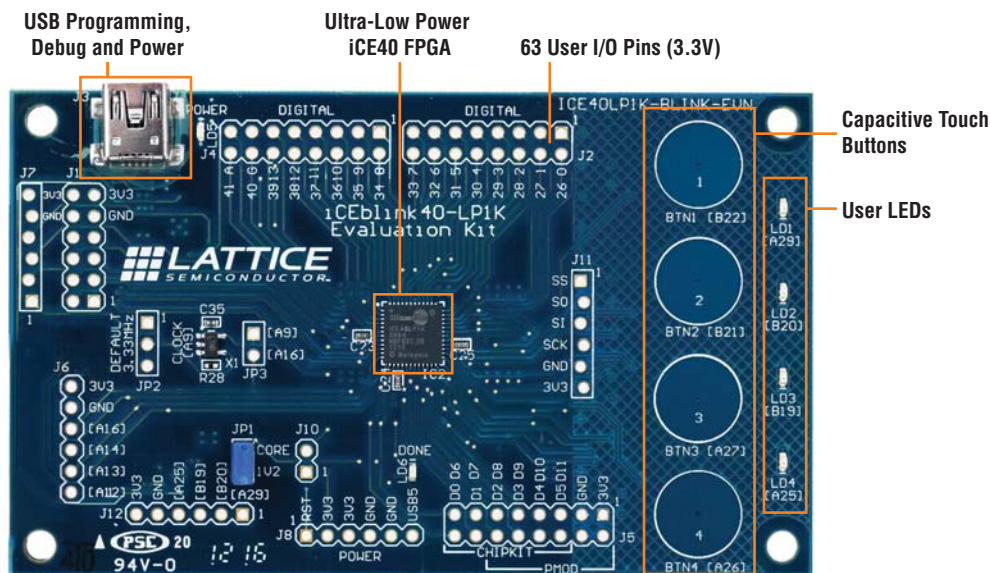
Thank you for choosing the Lattice Semiconductor iCEblink40™-LP1K Evaluation Kit.

This guide describes how to begin using the iCEblink40-LP1K Evaluation Kit, an easy-to-use platform for rapidly prototyping designs using the iCE40™ FPGA.

## Features

- Ultra low-power iCE40LP1K FPGA
- USB programming, debugging, virtual I/O functions, and power supply
- Four user LEDs
- Four capacitive-touch buttons
- 3.3 MHz clock source
- 1Mbit SPI serial configuration PROM
- Supported by Lattice iCEcube2™ design software
- 63 LVCMOS/LVTTL (3.3V) digital I/O connections on 0.1" through-hole connections
- Supports third-party I/O expansion boards and modules, including 3.3V Arduino Shield boards (requires additional sockets, not supplied)

**Figure 1. iCEblink40 LP1K Evaluation Board and Major Hardware Features**



## Software Requirements

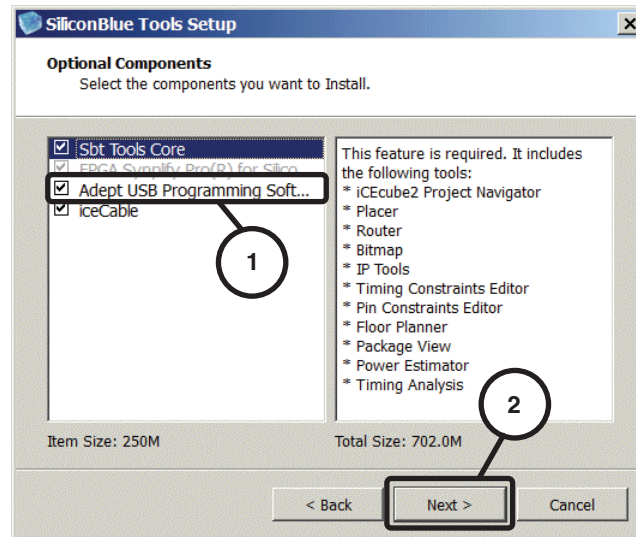
Before using the iCEblink40 board, please be sure to download and install iCEcube2 Release 2011.12 or later. This and later versions include the programming software for the iCEblink40 board. Currently, the programming software is only available for the Windows operating system.

<http://www.latticesemi.com/products/designsoftware/icecube2/downloads.cfm>

During the installation process, be sure to install the Adept USB Programming Software, as shown in Figure 2.

1. Make sure that **Adept USB Programming Software** is checked. This is the default setting.
2. Click **Next**.

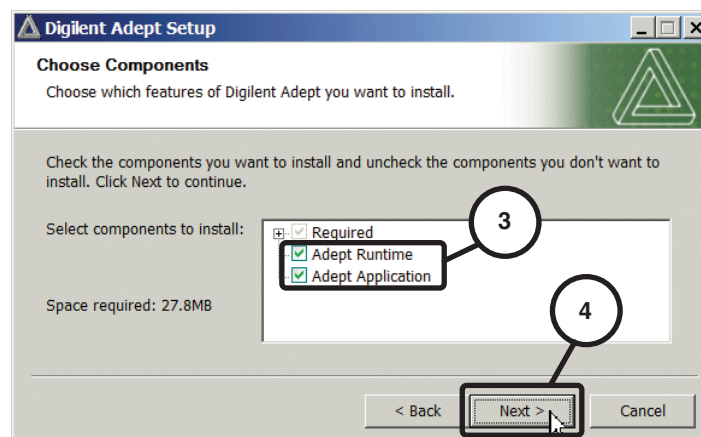
**Figure 2. Select the Adept Programming Software for Installation**



A few steps later, select the installation for the Adept programming software, as shown in Figure 3.

3. Make sure that both the **Adept Runtime** and **Adept Application** options are checked, which are the default settings.
4. Click **Next**.

**Figure 3. Adept Setup Options**



## Connecting to the iCEblink40 Evaluation Board

Before connecting the iCEblink40 board, be sure to download and install a supported version of the iCEcube2 software.

Connect the iCEblink40 evaluation board to your PC using the USB cable provided. The USB connector on the board is labeled with reference designator J3 and is located in the upper left corner. Once connected, the red

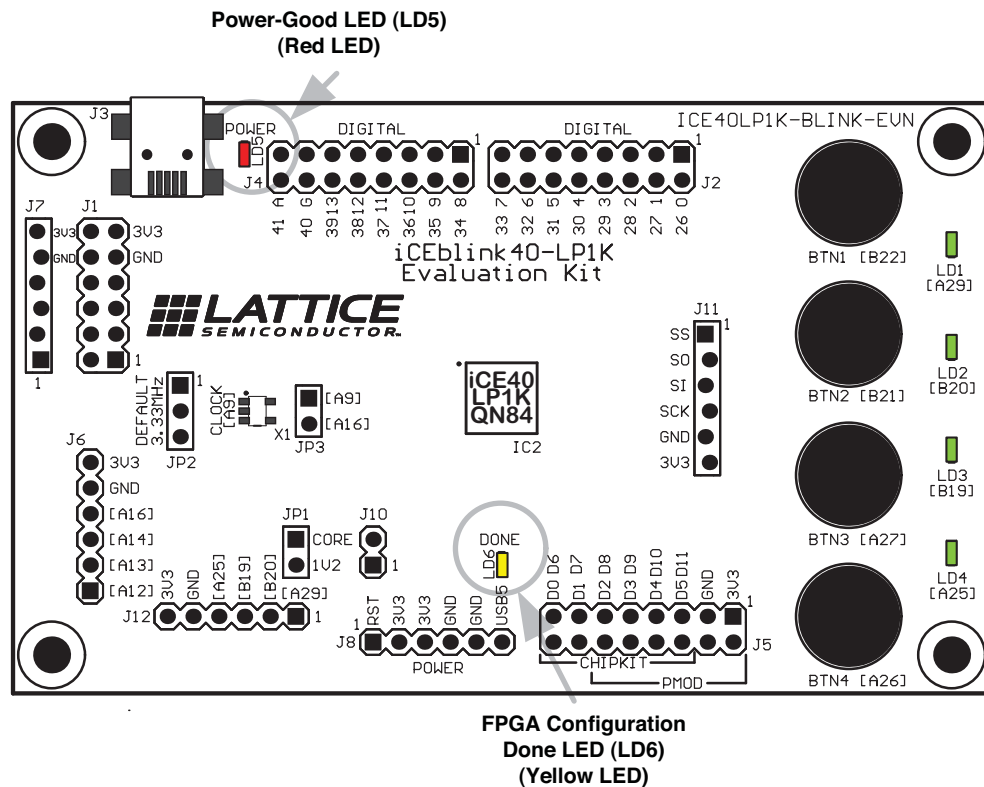
power-good LED (LD5) adjacent to the USB connector illuminates. See Figure 4 to locate the power-good LED.

## Power and Configuration Status LEDs

The iCEblink40 evaluation board has two status LEDs, as shown in Figure 4. These two status LEDs indicate the current status of the iCEblink40 board, as listed in Table 1. The red LED, LD5, located near the USB connector indicates if the USB power supply, the 3.3V supply, and the 1.2V supply are within the specified ranges.

The yellow LED, LD6, located below the FPGA indicates whether the FPGA is configured properly. This LED lights up when the FPGA is correctly loaded with a valid bitstream.

**Figure 4. iCEblink40 Status LEDs**



### Table 1. iCEblink40 Status LED Descriptions

Power-Good LED (LD5)	Configuration DONE LED (LD6)	Description
On	On	The board is powered, the FPGA successfully configured and the FPGA application is operating.
Off	Off	Board is unpowered. Connect the board to a computer USB port, a powered hub, or a USB-based wall plug.  If board is plugged in and previously operating, indicates that an SPI Flash programming operation is in progress
On	Off	The board is powered but the FPGA is not yet configured.  ACTION: Program the onboard SPI Flash PROM with a valid FPGA configuration bitstream.
Off	On	ERROR: The board is powered but there is a problem with the USB power supply or with the on-board regulator.

## Pre-programmed Demonstration Design

The iCEblink40 board comes preprogrammed with a demonstration application. The application supports two interfaces.

1. Control the LEDs from the four capacitive touch buttons on the board itself.
2. Control the LEDs and other internal logic using the USB-based I/O expansion interface.

## Operating the Capacitive Touch Buttons

Upon power up, the green LEDs on the board scroll in an upward pattern, as described in Figure 5. Pressing any of the capacitive touch buttons stops the LEDs from scrolling and places the board in a different operating mode.

**Figure 5. Preprogrammed Demonstration Design**

**LEDs Scroll Upward**



**Toggle LED with Button Press**



1. Power On  
Green LEDs scroll upward.
2. Press any button to enter LED Toggle Mode.
3. Press a button to toggle the associated LED on or off.  
If no button is pressed within five seconds, the board returns to Scroll LEDs Mode.

In the second operating mode, toggle individual LEDs on and off by pressing the associated capacitive touch button.

If no button was pressed during the last five seconds, the board returns to scrolling the LEDs.

The demonstration application is available for download from the Lattice website at:

[www.latticesemi.com/iceblink40-LP1K](http://www.latticesemi.com/iceblink40-LP1K).

## Virtual I/O Expansion Debugging Interface

The iCEblink40 board is powered and programmed via the USB interface. Additionally, the USB interface also provides a convenient means to monitor and control logic inside the FPGA, as shown in Figure 6. The USB controller drives a byte-wide parallel port expander implemented within the FPGA, controlled by software running on the PC. The Digilent ADEPT2 I/O Expansion screen, shown in Figure 7, provides a mix of virtual switches, pushbuttons, LEDs, light bars, and 32-bit input and outputs.

Figure 6. iCEblink40 Board Supports Virtual I/O Connections over USB

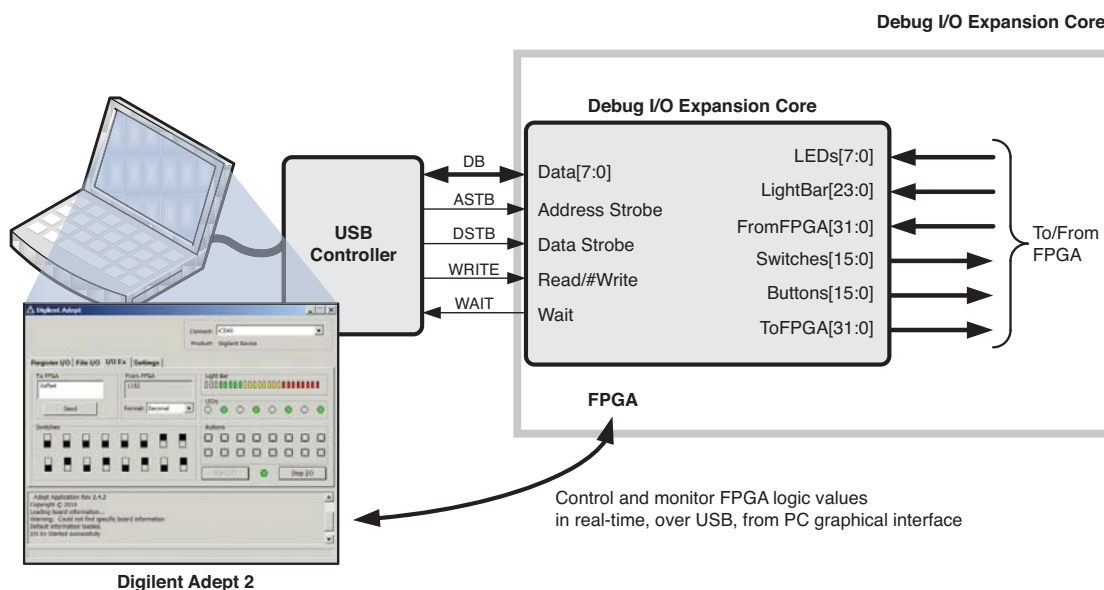
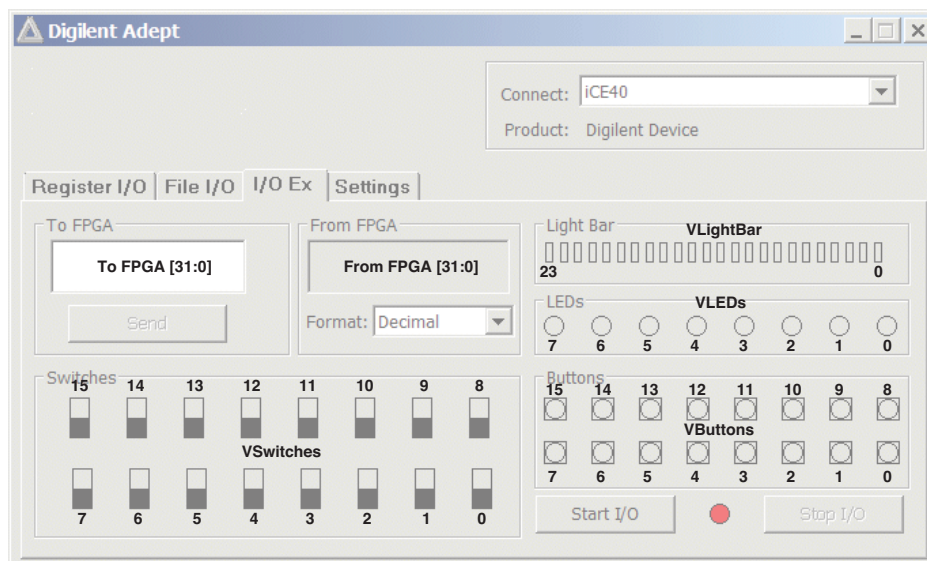


Figure 7. Digilent Adept 2 I/O Expansion Interface and FPGA Connections



## Using the Virtual I/Os in the Demo Application

By default, the virtual I/Os are disconnected and the USB controller's I/O connections to the FPGA are high-impedance (Hi-Z).

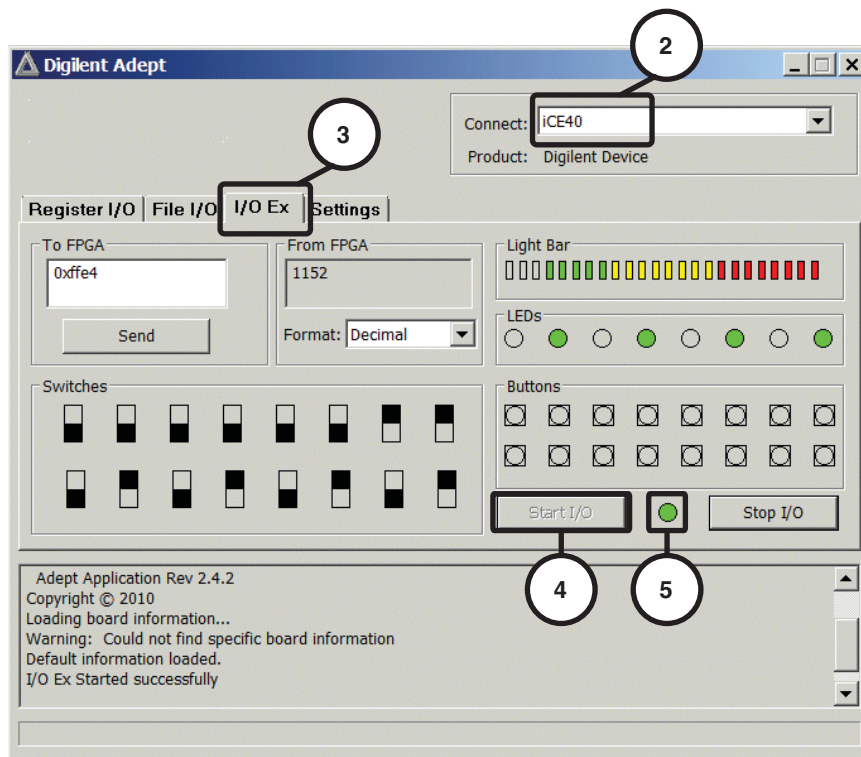
To connect the virtual I/O, perform the following steps outlined in Figure 8.

1. From the Windows Start menu, select **Start > All Program > Digilent > Adept > Adept**.
2. Ensure that the Adept interface connects to the iCE40.
3. Click the **I/O Ex** tab.



4. Click **Start I/O**. Remember, the associated I/O Expander design must be part of the compiled FPGA design before the Virtual I/Os work.
5. If the virtual I/O expansion design is functioning correctly, the green virtual status LED will turn from red to green.

**Figure 8. Starting the Digilent Adept Virtual I/O Expansion Application**



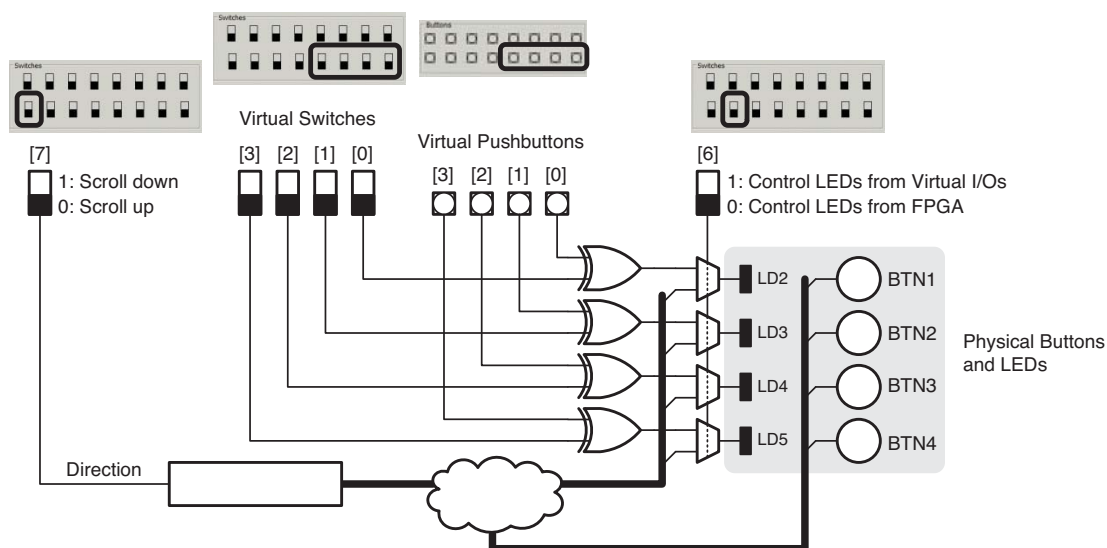
To disconnect the virtual I/O interface, simply click the **Stop I/O** button in the graphical interface.

## Controlling the Physical LEDs from Virtual I/Os in the Demonstration Design

When active, the virtual I/Os optionally control the physical LEDs on the board, as shown in Figure 9. For example, with the virtual I/Os active, change the position of virtual switch [7] (the bottom left switch in the graphical interface). Note how the physical LEDs on the board change direction.

Change virtual switch [6] to the up position. Now, the physical LEDs are controlled by the virtual switches [3:0] and virtual pushbuttons [3:0]. The values of the virtual switches are XORed together inside the FPGA. The virtual slide switches set a specific value for the physical LEDs. The pushbutton momentarily inverts the value while the virtual pushbutton is pressed in the graphical interface.

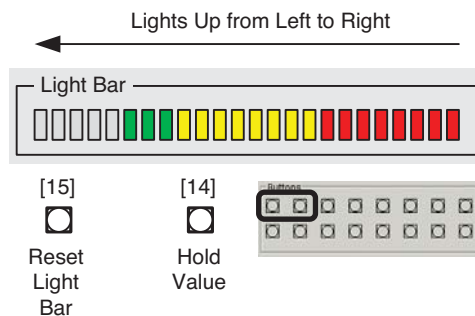
**Figure 9. Controlling the Physical LEDs from Virtual I/O**



## Controlling the Virtual Light Bar in the Demonstration Design

When the virtual I/Os are active, the virtual light bar lights up from left to right, controlled by logic inside the FPGA. The virtual pushbuttons [15] and [14] control the light bar. Pushbutton [15] resets the light bar, clearing all the lights. Pushbutton [14] forces the light bar to hold its current value.

**Figure 10. Controlling the Virtual Light Bar**



## Controlling the Virtual LEDs in the Demonstration Design

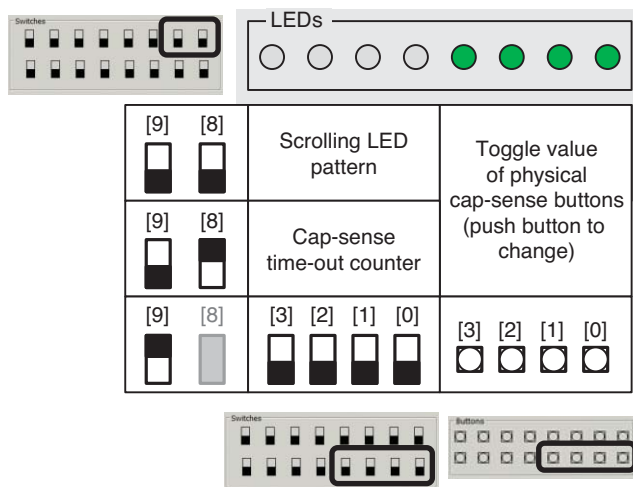
The virtual I/O interface includes eight, round, green LEDs, as shown in Figure 11. The values displayed on these virtual LEDs depends on the settings of virtual switches [9] and [8].

The eight LEDs are separated into left and right halves. When both virtual switches [9] and [8] are Low—the down position—the left LEDs echo the scrolling pattern of the LEDs, regardless if a physical cap-sense button was pressed. Use virtual switch [7] to reverse the direction of these LEDs. The right-most LEDs show the current toggle status of the four physical cap-sense buttons.

Changing virtual switch [8] to High—the up position—the four left-most LEDs then show the current value of the time-out counter than marks the five seconds after pressing a cap-sense button. Press a cap-sense button to reset the timer and note that the toggle status of the physical button changes on the right-most LEDs. The timer resets each time a physical button is pressed. Wait five seconds and the physical LEDs change back to the scrolling pattern.



Figure 11. Controlling the Virtual LEDs

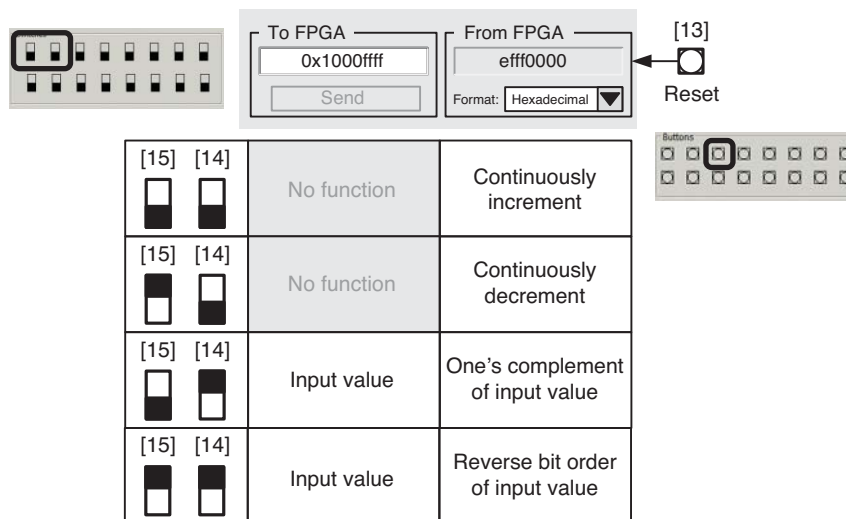


When virtual switch [9] is High—in the up position—then the left LEDs are controlled by virtual switches [3:0] and the right LEDs are controlled by virtual pushbuttons [3:0].

## Virtual Values to and from FPGA in the Demonstration Design

The virtual I/O interface also includes a 32-bit value from the FPGA logic and a 32-bit value to the FPGA logic, as shown in Figure 12. Two virtual switches, [14] and [15], control the behavior in of the virtual 32-bit values in the demonstration design.

Figure 12. Virtual Values to and from the FPGA



## Clocking Resources

The iCEblink40 board includes a Linear Technology LT1799 oscillator (X1 on the board and in the schematic) to generate a 3.33 MHz clock.

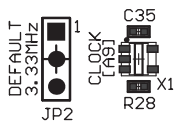
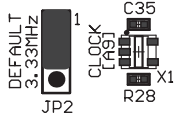
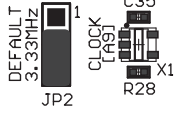
## FPGA Input

The output from the LT1799 oscillator feeds pin A9 of the iCE40LP1K FPGA. FPGA pin A9 is also the global buffer input GBIN7.

## Supporting Other Frequencies

On the iCEblink40 board, the LT1799 produces a 3.3 MHz clock output by default. Other frequencies are possible via simple modifications of the board using the 1x3 connections on JP2, as listed in Table 2.

**Table 2. Selecting Other Oscillator Frequencies Using Jumper JP2**

Clock Frequency	JP2 Setting	Jumper Position
3.33 MHz (default)		None
333 kHz		Upper Position
33.3 MHz		Lower Position

## User LEDs

The iCEblink40 iCE40LP1K evaluation board includes four green user LEDs, located along the left side of the board, as shown in Figure 1.

## Operation

To light a user LED, drive the associated FPGA pin High, as shown in Table 3. To darken the LED, drive the associated FPGA pin Low.

**Table 3. User LED Operation**

Operation	FPGA Action
Light LED	Drive High (1)
Darken LED	Drive Low (0)

The LEDs may appear to glow slightly before the FPGA is configured or if the FPGA pin is unused. This is because the FPGA I/Os have a soft pull-up resistor which may provide just enough current for the LED to glow dimly. To completely turn off an LED, drive it Low.

## FPGA Connections

The FPGA drives the user LEDs using the FPGA pins listed in Table 4. These same signals also connect to the J12 header located in the lower left corner.

**Table 4. User LED Connections**

Designator	Location	FPGA Pin	Header Connections
LD1	LD1 [A29]	A29	J12.1
LD2	LD2 [B20]	B20	J12.2
LD3	LD3 [B19]	B19	J12.3
LD4	LD4 [A25]	A25	J12.4

## Capacitive Touch Buttons

The iCEblink40 iCE40LP1K evaluation board has four capacitive-touch buttons, located toward the left side of the board, as shown in Figure 1. These buttons have dedicated connections only to the FPGA. These signals go nowhere else on the board and are not available on any of the breakout headers.

## FPGA Connections

Table 5 lists the four capacitive touch buttons on the iCEblink40 board and the associated FPGA pins.

**Table 5. Capacitive Touch Buttons**

Designator	Location	FPGA Pin
BTN1	BTN1 [B22]	B22
BTN2	BTN2 [B21]	B21
BTN3	BTN3 [A27]	A27
BTN4	BTN4 [A26]	A26

## Operation

Figure 13 shows the circuit used for each capacitive-touch button. Each button is attached to one I/O pin on the FPGA. Each signal line includes a 100 k $\Omega$  pull-up resistor to 3.3V and a 100 pF capacitor down to ground.

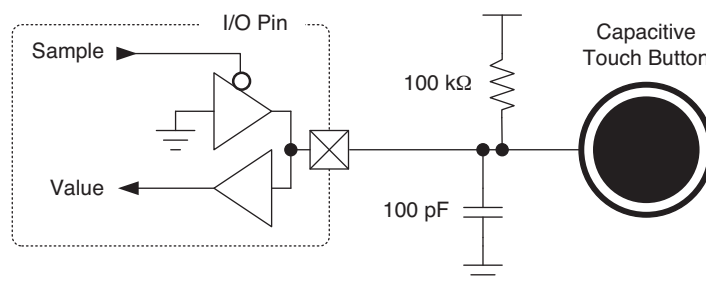
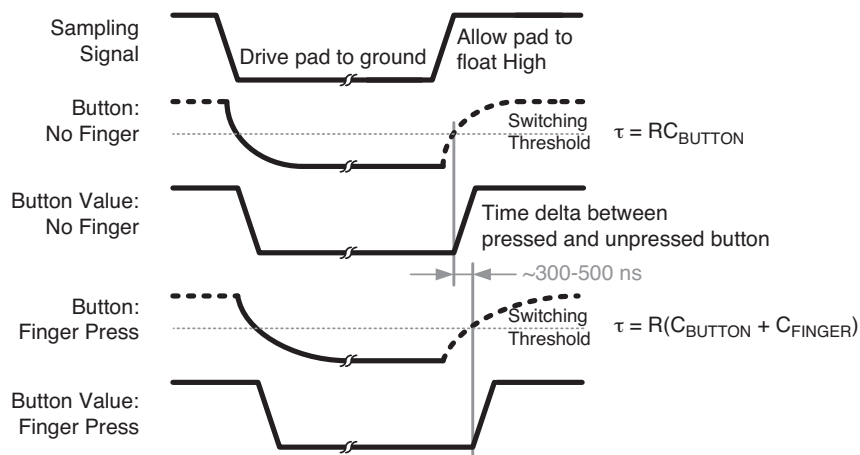
**Figure 13. Example Capacitive Touch Button Circuit**


Figure 15 shows the general overall flowchart for the demonstration design to read the value on a capacitive touch button.

The sampling signal drives the voltage on the capacitive-touch button to ground in order to bleed of any residual charge as shown in Figure 14. After a period of time, depending on the button sample frequency, the button is allowed to float High.

Once the FPGA output goes to Hi-Z (high-impedance, floating, three-state), the 100k $\Omega$  pull-up resistor to 3.3V charges the 100 pF capacitor. After about an RC time constant ( $\tau$  or tau), the voltage on the pad exceeds the input switching threshold of the FPGA. A finger pressed against the capacitive-touch button adds about another 5 pF of capacitance, increasing the RC constant and delaying the Low-to-High transition for a pressed button.

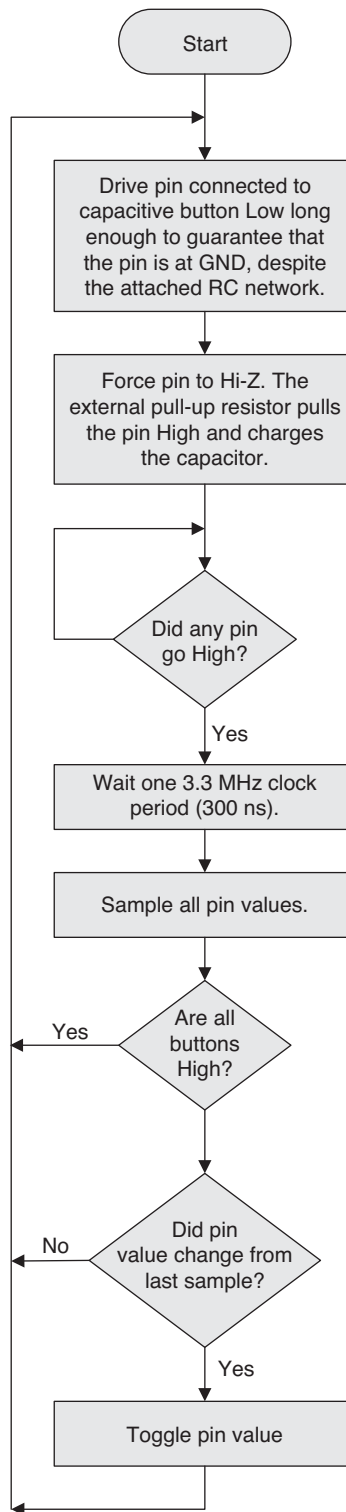
**Figure 14. Capacitive Touch Timing Examples**



The switching time difference between an unpressed and one or more pressed buttons is roughly 300 to 500 ns. Using the 3.33 MHz input, this amounts to a one clock delay difference between an unpressed and pressed buttons.

The simple circuit used on the iCEblink40 board detects simultaneous button presses on up to three of the capacitive-touch buttons. Pressing all four buttons is the same as pressing no buttons.

Figure 15. iCEblink40 Demo Application Capacitive Touch Button Flowchart

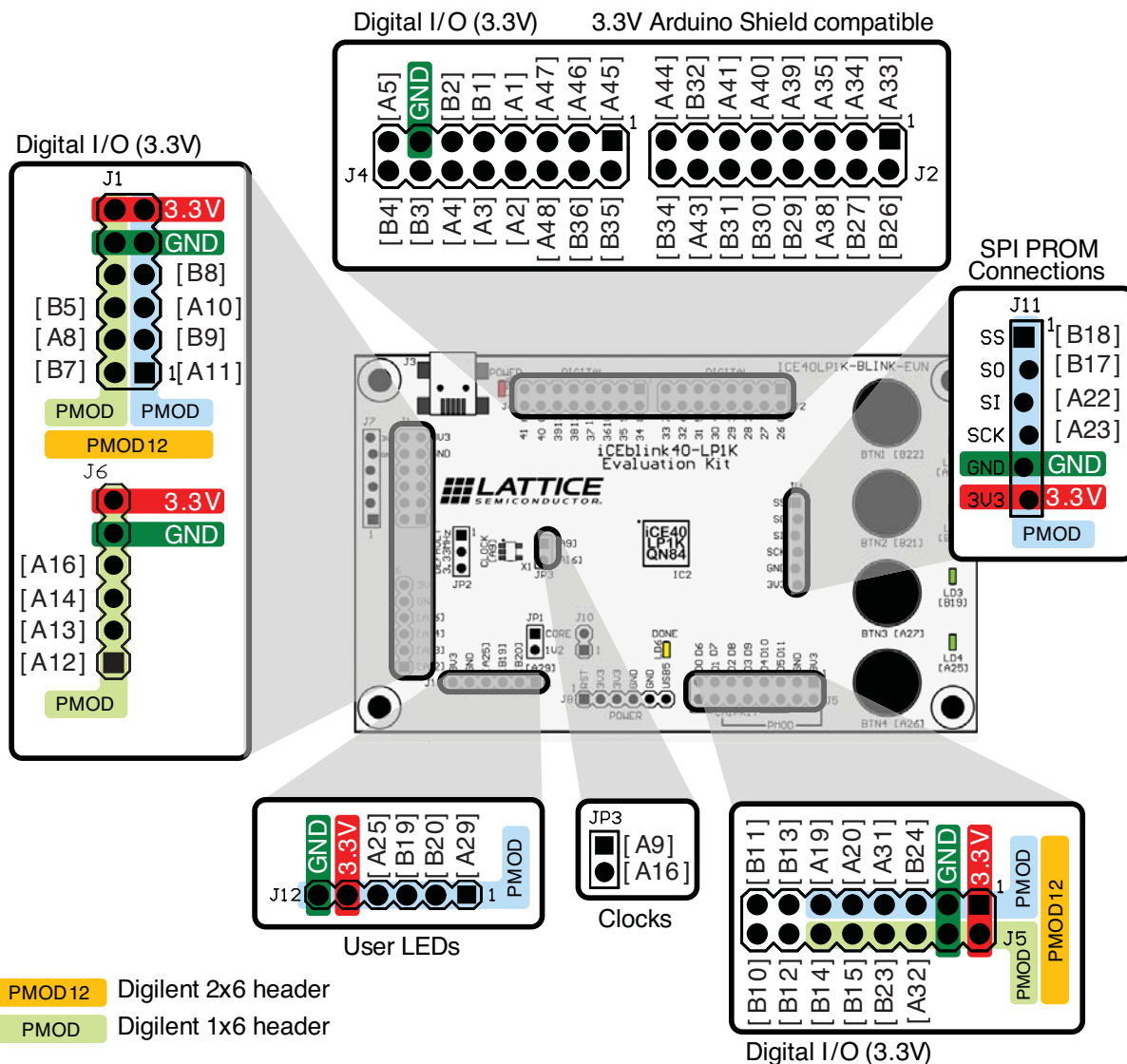




## User I/O Connections

Figure 16 shows the location of the 3.3V-compatible digital I/O connections on the iCEblink40 board. Each connection shows the pin number of the FPGA I/O pin that attaches to the connection. Likewise, Table 6 lists the various I/O headers and their designed usage.

**Figure 16. Location of the 3.3V Digital I/O Connections and the FPGA Pin Number**



**Table 6. Digital I/O Headers and Their Functions**

I/O Header Group	Header Type	Location	Function
J2	2x8 0.1" centers	Top edge, middle	3.3V digital I/O. Compatible with 3.3V Arduino Shield boards.
J4	2x8 0.1" centers	Top edge, left side	3.3V digital I/O. Compatible with 3.3V Arduino Shield boards.
J1	2x6 0.1" centers	Left edge, top	3.3V digital I/O. 3.3V digital I/O. Compatible with Digilent 1x6 and 2x6 PMod modules. Also supports double PMod12 modules when used with header J6.
J6	1x6 0.1" centers	Left edge, bottom	3.3V digital I/O. Compatible with Digilent 1x6 PMod modules.
J7	1x6 0.1" centers off-set	Left edge, top	Production programming of USB controller.
J11	1x6 0.1" centers off-set	Middle, toward left	3.3V digital I/O. Connections between the mobile FPGA and the SPI PROM. Compatible with Digilent 1x6 PMod modules.
J5	2x8 0.1" centers	Bottom edge, right side	3.3V digital I/O. Portions compatible with Digilent 1x6 and 2x6 PMod modules. Portions also compatible with 3.3V Arduino Shield boards.
JP3	1x2 0.1" centers	Middle, to left of FPGA	3.3V digital I/O. Clock connections from the LTC1799 oscillator (GBIN7) and possible into GBIN2.
J12	1x6 0.1" centers	Bottom edge, left side	3.3V digital I/O. Connections to the user LED I/O. Compatible with Digilent 1x6 Pmod modules.

## Supported Pmod Peripheral Modules

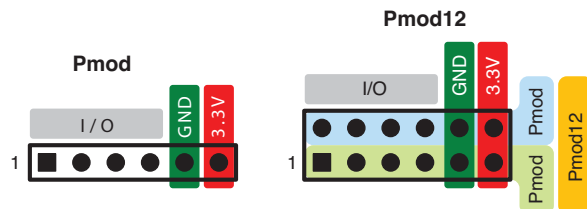
As shown in Figure 16, the iCEblink40 board supports a variety of Pmod peripheral modules for easy I/O expansion. Table 7 lists the 0.1" through-hole headers on the iCEblink40 board that support Pmod modules. Pmod modules come in a few different form factors and each Pmod header includes power and ground supplies. Figure 17 shows the how the different Pmod form factors interrelate. The easiest way to support a Pmod module is to add the appropriate female socket listed, or an equivalent. Straight-through or right-angle through-hole sockets are listed. Male headers are also possible solutions when using the interface cable provided with most Pmod modules.

**Table 7. Pmod Module Headers**

Header	Type	Female Socket (Manufacturer/Part Number)	
		Straight-through	Right-angle
J6, J12	1x6 header on 0.1" centers. A six-pin Pmod header.	Sullins Connector Solutions PPPC061LFBN-RC	Sullins Connector Solutions PPPC061LGBN-RC
J1	2x6 header on 0.1" centers. A Pmod12 header that also supports two six-pin Pmod modules.	Sullins Connector Solutions PPPC062LFBN-RC	Sullins Connector Solutions PPPC062LJBN-RC
J5	2x8 header on 0.1" centers. The left side of header J5 forms a Pmod12 header, as shown in Figure 16. A 2x6 header similar to J1 can also be used but must be mounted toward the right end of the holes as marked.	Sullins Connector Solutions PPPC082LFBN-RC	Sullins Connector Solutions PPPC082LJBN-RC

As shown in Figure 17, a Pmod module has six connections—four I/O plus power and ground. A Pmod12 module has 12 connections and the module is effectively two six-pin Pmod modules stacked together. Most of the Pmod modules also include interface cables to allow easy connection to other header types.

Figure 17. Pmod Module Types



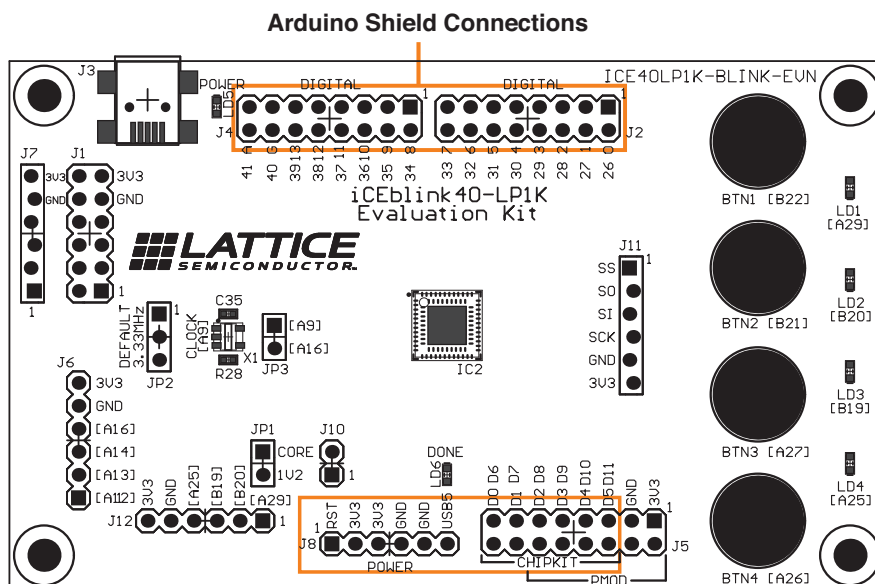
For a complete list of Pmod peripheral modules, visit the Digilent web site.

[www.digilentinc.com/Products/Catalog.cfm?NavPath=2.401&Cat=9](http://www.digilentinc.com/Products/Catalog.cfm?NavPath=2.401&Cat=9)

## Arduino Shield Board Support

The iCEblink40 board also mechanically and electrically supports select 3.3V Arduino Shield boards popular in the microcontroller development community. The Shield connections are located on headers J4, J2, J8 and a portion of J5 as shown in Figure 18. Headers jumper J4 and J2 are 3.3V digital I/O connections. Header J8 provides power connections to the Shield board. The left side of header J5 also provides 3.3V digital I/O but the 3.3V and GND connections do not connect to the Shield board.

Figure 18. Arduino Shield Board Connections



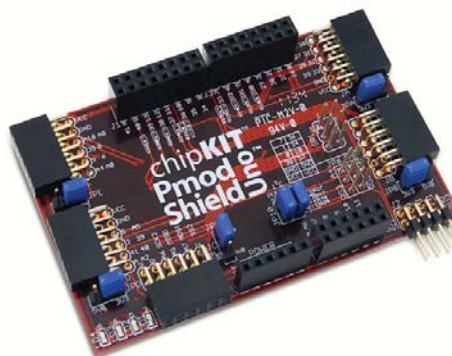
## Tested Arduino Boards

Figure 19 shows the Arduino Shield boards have been tested for basic compatibility. Other Arduino Shield boards may also be compatible.

**Figure 19. Compatible Arduino Shield Boards**



chipKITBasic I/O Shield



chipKIT Pmod Shield-Uno

## USB Interface

The iCEblink40 board is powered by connecting the board to a computer USB port, a power USB hub, or a USB-based AC adapter, commonly used in consumer electronics. A typical USB port provides up to 500 mA at 5V, providing up to 2.5W of total power. The iCE40LP1K consumes SIGNIFICANTLY LESS power, even when operating at full performance. However, be careful when using the board to power off-board peripheral funds.

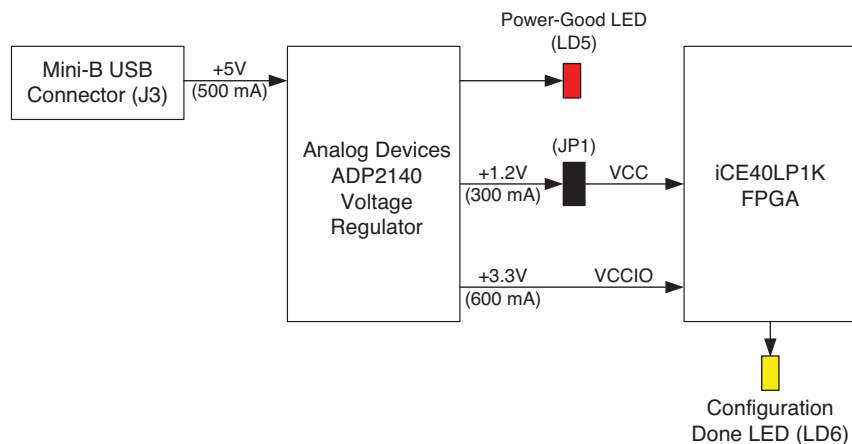
## Connector

The USB connector to the board is located in the upper left corner, labeled J3. The board connects using a standard USB cable with a male mini-B connector.

## Power Supply

Figure 20 shows the iCEblink40 power supply circuit that derives power from the USB mini-B connector (J3). The USB connector provides up to 500 mA at +5V DC. An Analog Devices ADP2140 regulator generates +1.2V for the FPGA core VCC and +3.3V for all I/O connections. The regulator also indicates when power is good and lights up the red power-good LED (LD5).

**Figure 20. iCEblink40 USB Power Supply Circuit**



Jumper JP1 provides a convenient location from which to measure core power to the FPGA.

## SPI Flash Programming

The USB interface also provides Flash programming for the on-board SPI PROM, as described in “Programming the iCEblink40 Board” below.

### Diligent Parallel Port (DPP)

The Diligent Parallel Port (DPP) interface is used for virtual I/O and debugging using a USB connection to the board from a Windows PC. See “Virtual I/O Expansion Debugging Interface” on page 5 for additional information.

## 1Mbit SPI Configuration PROM

The configuration bitstream for the iCE40 FPGA is stored in a M25P10A 1Mbit SPI serial Flash PROM. The PROM is large enough to hold two configuration images and supports the iCE40 WarmBoot feature, if so enabled within the FPGA application. The PROM is physically located on the back side of the board.

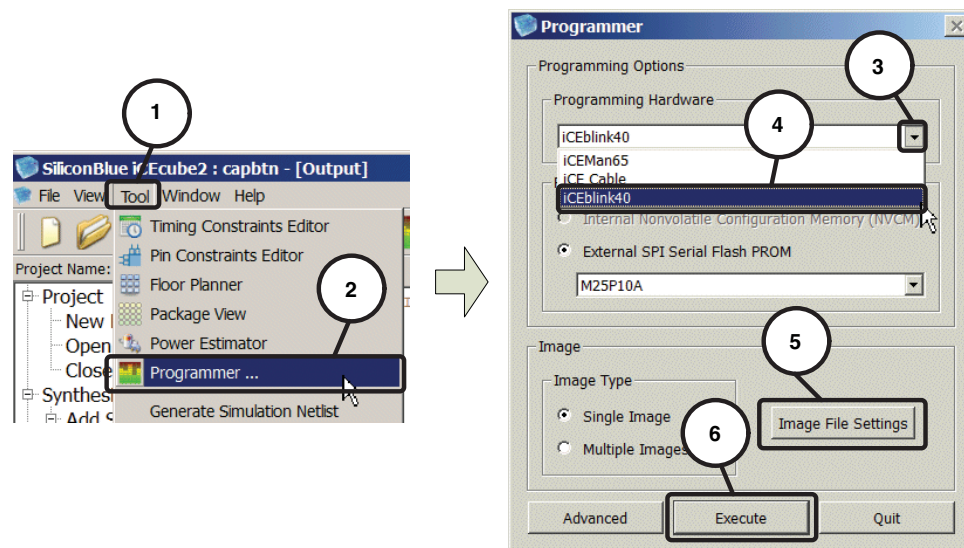
## Programming the iCEblink40 Board


The iCEblink40 board includes on-board USB-based programming support either from the Lattice iCEcube2 software or using a command from a console window or DOS box.

### From iCEcube2

Figure 21 shows the command sequence for programming the SPI Flash PROM on the iCEblink40 board using the iCEcube2 development software.

**Figure 21. Programming the iCEblink40 Board from iCEcube2**



1. Select **Tool > Programmer** from the iCEcube2 menu bar.
2. Click the dropdown button (  ) under **Programming Hardware**.
3. Select **iCEblink40**.
4. The bitstream file should already be set appropriately based on the iCEcube2 project settings. If not, click **Image Files Settings** to select the configuration bitstream file.
5. Click **Execute** to program the iCEblink40 board.



If all is working correctly, the power-on LED and the configuration done LED will both go out momentarily as iCEcube2 programs the on-board SPI Flash PROM. After programming is complete, both LEDs should light up again and the FPGA will execute the new configuration image.

## From Command Line

The iCEblink40 programming software can also be executed from a console window or DOS box. To open a console window or DOS box, click the **Start** button and type **cmd** in the textbox immediately above the Start button.

### Executable Location

After installation, the programming software executable is called **iceutil.exe** and is located in the **\Sbt-Tools\sbt\_backend\bin\win32\opt** directory. The iceutil.exe executable can be copied into the same directory as the FPGA bitstream image or can be pointed to on the command line.

### FPGA Bitstream Configuration File

The required bitstream image is part of the iCEcube2 project. Multiple versions of the bitstream are stored in the **<projname>\_Implmnt\sbt\outputs\bitmap** directory. The raw hexadecimal version of the bitstream is called **<projname>\_bitmap.hex**. The alternate format of the same information is an Intel hexadecimal file called **<projname>\_bitmap\_int.hex**.

### Raw Hexadecimal Command Example

```
<path>/iceutil -d iCE40 -res -cr -m M25P10A -fh -w <path/projname>_bitmap.hex
```

### Intel Hexadecimal Command Example

```
<path>/iceutil -d iCE40 -res -cr -m M25P10A -fi -w <path/projname>_bitmap_int.hex
```

### Help

```
<path>/iceutil -help
```

## Testing Core Power

Jumper JP1 provides the ability to measure core power consumption by the FPGA. Two power measurement methods are supported.

### Easy Method Using a Multimeter

Connect the iCEblink40 board through your high-accuracy multimeter. Use a meter with a minimum of 10,000 counts; 50,000 counts or more is recommended for better accuracy.

To take a quick measurement, follow these steps.

1. Disconnect power to the iCEblink40 board by removing the USB cable connection, either at the board or at the computer.
2. Remove the jumper JP1, which isolates the FPGA's core supply from the 1.2V supply on the board.
3. Connect your multimeter's alligator or test clips to the stake pins on header JP1.
4. Configure the multimeter to measure current using its highest mA or Amp range. This setting typically has the lowest voltage drop internally within the meter.
5. Re-connect the USB cable that supplies power to the iCEblink40 board and configure the FPGA device if necessary.
6. Observe the power reading on the multimeter. At low clock rates, which results in lower power consumption, switch the meter to a lower amperage setting for better accuracy. However, this also may increase the resistance across the meter leads. Using too low of a meter setting causes a large voltage drop within the meter, potentially violating the minimum input voltage specification to the FPGA device.

7. The value measured by the multimeter is a current. Convert the measurement to power using Equation 1. The voltage is the operating voltage, the voltage across the jumper. This value can be accurately measured with a second multimeter to show the voltage drop across the first. However, just measuring the initial voltage, before taking any current readings, usually provides acceptable accuracy and the voltage drop across the meter is generally small.

$$\text{Power} = \text{Current} \times \text{Voltage} \quad (1)$$

Although this method is easy, here are a few caveats and pointers.

- **Always start at the highest current setting for your meter. Using too small a setting may damage your meter!** After determining the maximum current range for your measurement, then you can safely use the appropriate lower current setting.
- The voltage drop across the meter leads may violate the minimum supply voltage specification for the mobile-FPGA device. To determine the voltage drop, use a second multimeter to measure either the voltage across the first meter's leads during a test or the resistance between the first meter's leads.
- Using the highest current measurement setting typically results in the lowest voltage drop.

### Using High-Precision, Small-Value Resistors

For more-accurate, time-sensitive measurements, place a low-value resistor across the jumper test point. According to Ohm's Law, the current passing through the resistor produces a voltage drop. Measure the voltage differential across the resistor during expected operation. Convert the measurement to power using Equation 2. The voltage is the measured voltage across the resistor; the resistance is the value of the resistor.

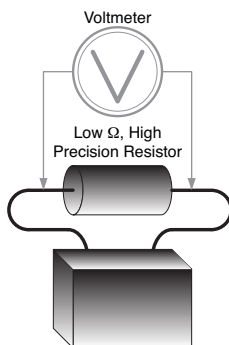
$$\text{Power} = \frac{(\text{Voltage})^2}{\text{Resistance}} \quad (2)$$

The following are a few guidelines on selecting a resistor.

- Use a high-precision resistor.
- The resistor must handle the power dissipated under the anticipated test conditions.
- Too small a resistor value may result in too small a voltage difference across the resistor to measure with your test equipment.
- Too large a resistor value may result in too large of a voltage difference across the resistor. Too large a voltage drop might violate the minimum voltage specifications for the FPGA device.

Figure 22 shows an example header block designed to fit over one of the jump locations. Measure the voltage drop across the low-value resistor, either with a voltmeter or with data acquisition equipment.

**Figure 22. Resistor Header Block**

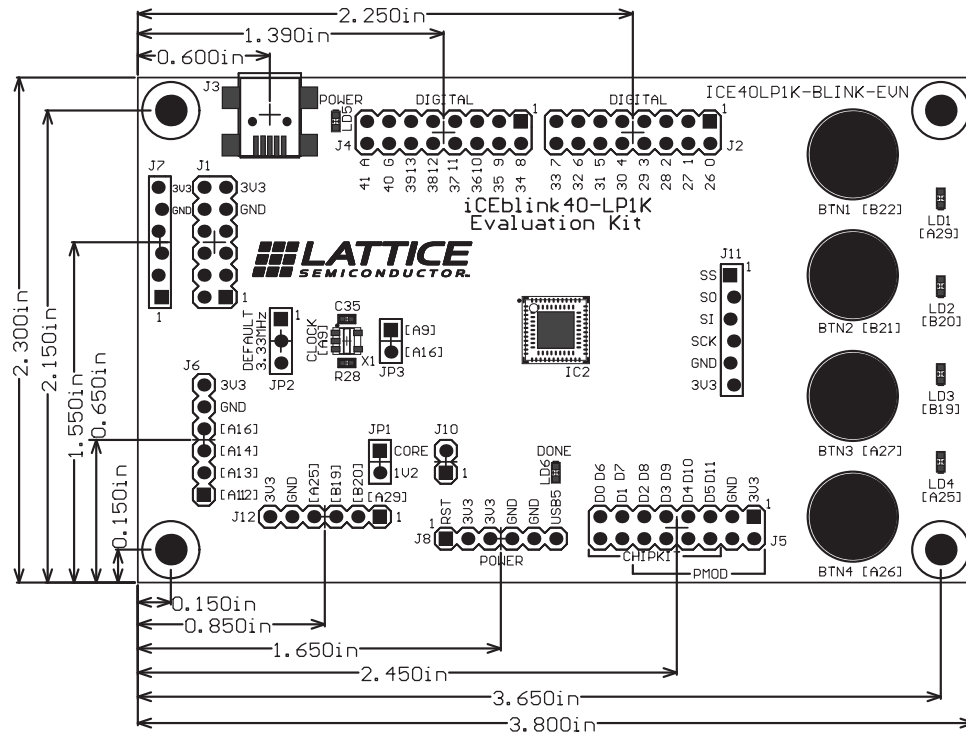


This method is recommended for taking power measurements over time.


## Mechanical Specifications

Figure 23 shows the mechanical dimensions for the iCEblink40 board, including the location of the four mounting holes. With a jumper installed on JP1, the board height is approximately 0.700 inches high, including the four rubber feet mounted on the bottom side of the board.

**Figure 23. iCEblink40 LP1K Board Mechanical Dimension**



## Ordering Information

Description	Ordering Part Number	China RoHS Environment-Friendly Use Period (EFUP)
iCEblink40-LP1K Evaluation Kit	ICE40LP1K-BLINK-EVN	

## Technical Support Assistance

Hotline: 1-800-LATTICE (North America)  
+1-503-268-8001 (Outside North America)  
e-mail: [techsupport@latticesemi.com](mailto:techsupport@latticesemi.com)  
Internet: [www.latticesemi.com](http://www.latticesemi.com)

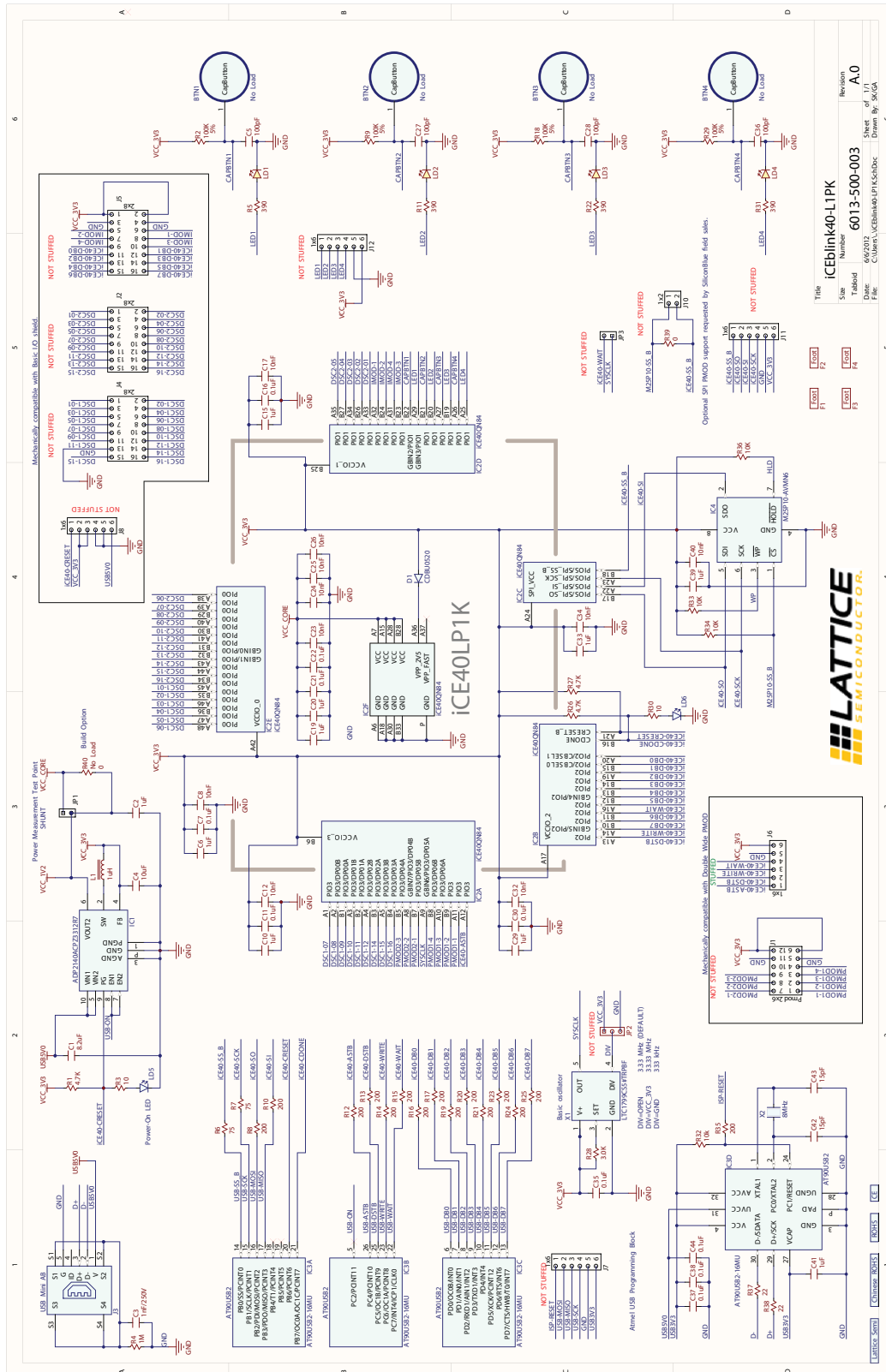
## Revision History

Date	Version	Change Summary
July 2012	01.0	Initial release.
September 2012	01.1	Nomenclature change from “mobileFPGA” to “FPGA”.

© 2012 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at [www.latticesemi.com/legal](http://www.latticesemi.com/legal). All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

## Appendix A. Schematic

Figure 24. Schematic



---

## Appendix B. Bill of Materials (Major Components)

**Table 9. Bill of Materials**

Reference	Vendor	Part Number	Description
IC2	Lattice Semiconductor	<a href="#">iCE40LP1K-QN84</a>	iCE40 LP-series ultra low-power FPGA
IC1	Analog Devices	<a href="#">ADP2140ACPZ3312R7</a>	Low-quiescent buck/LDO regulator (1.2V, 3.3V)
X1	Linear Technology	<a href="#">LTC1799CS5#TRPBE</a>	Oscillator
IC4	Micron Technology	<a href="#">M25P10-AVMN6</a>	1Mbit SPI serial configuration Flash PROM
IC3	Atmel Corporation	<a href="#">AT90USB162-16MU</a>	USB programming and debugging interface