iCEdragon Demo Board

User's Guide

November 2012
EB80_01.0

Introduction

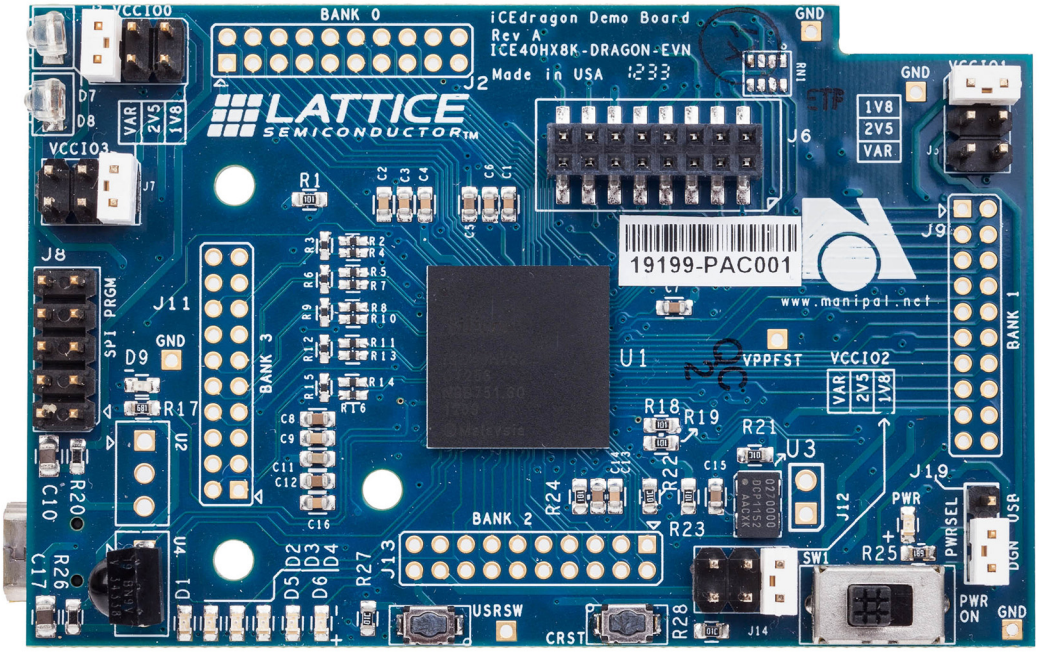
Thank you for choosing the iCEdragon™ Demo Board. This board is designed to help you develop hardware solutions based on the Android OS using the iCE40™ ultra-low density FGPA.

To get the latest news, updates and downloadable resources for the iCEdragon Demo Board – including demos and documentation, visit the Lattice web site at www.latticesemi.com/boards and follow the links to the iCEdragon Demo Board.

Features

- iCE40-HX8K FPGA in 256-ball caBGA package
- Generous I/O access on PCB
- Six status LEDs
- Two pushbuttons
- USB mini plug
- IR Sensor
- Two IR transmitters
- On/off switch
- 8 Mbit SPI serial configuration PROM
- USB mini cable included

Figure 1. iCEdragon Demo Board



The following items are not included with the iCEdragon Demo Board, but may be required to fully utilize the board in some situations:

- Programming Cable: Lattice HW-USBN-2A may be used.
- DragonBoard development platform: See below for more details

Software Requirements

The following software is available for download from the Lattice web site. Go to: www.latticesemi.com/software and follow the appropriate links.

- iCEcube2™: For development with the iCE40 FGPA
- Programmer: For programming the iCE40 FPGA with a Lattice programming cable (cable available separately)

In addition, drivers and development software may be required for development with the Andriod OS. Specific details will depend on the level of development you intend to complete.

DragonBoard

The iCEdragon Demo Board is designed exclusively for use with the DragonBoard development system, available separately from BSquare. The DragonBoard is a system to aid in development of hardware systems based on the Android OS. The iCEdragon Demo Board is not designed for stand-alone operation; the DragonBoard development system is required to both power and program the iCEdragon Demo Board, as well as to run the demonstration(s) described in this document.

For more information on the DragonBoard kits and options, visit the BSquare website at:
www.bsquare.com/store-products.aspx

iCEdragon Demo Board Default Jumper Settings

Table 1. iCEdragon Demo Board Default Jumper Settings

Jumper	Parameter	Setting
J3	VCCIO0	Set at VCCVAR (the voltage on Bank 0, where IR Tx and Rx parts are connected).
J5	VCCIO1	Set at 1V8. This is the voltage on Bank 1 where the DragonBoard interfaces are wired. Must be set to 1V8 for compatibility with DragonBoard levels.
J19	PWRSEL	Set at DGN. Indicates the source of power, will be 2.85V from the DragonBoard's J6 connector for DragonBoard-related demos. This renders VCCVAR at 2.85V throughout the board.
J14	VCCIO2	Set at VCCVAR so that LEDs can light as required.
J7	VCCIO3	Set at 1.8V. No connections for current demos, set it at the lowest voltage.

Qualcomm IR-I^{2}C Bridge Demo Setup

The iCE40 FPGA on the iCEdragon Demo Board comes pre-programmed with the Qualcomm IR-I^{2}C Bridge Demo described below. The programming file and source code for this demo can be downloaded from the Lattice website at www.latticesemi.com/boards (navigate to the iCEdragon Demo Board page).

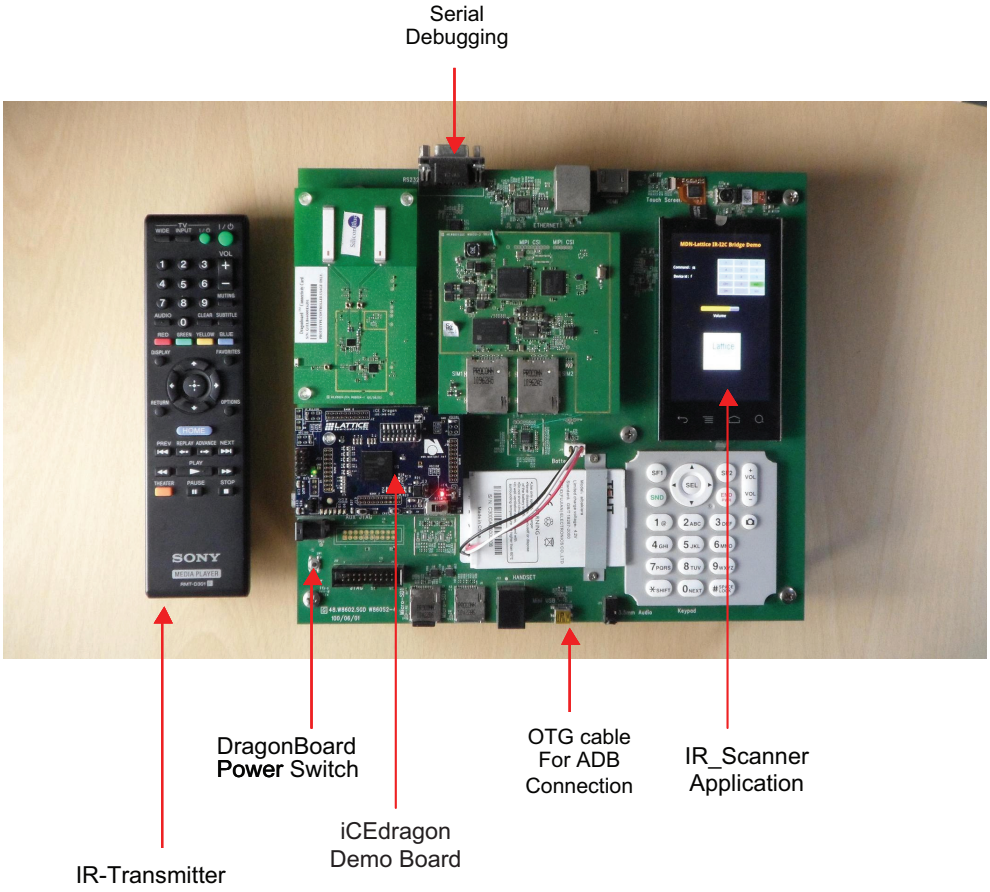
The Qualcomm IR-I^{2}C Bridge Demo setup consists of the iCEdragon Demo Board which includes an IR-Receiver interfaced with an iCE40 ultra-low density FPGA. The iCEdragon Demo Board is interfaced with a J6 connector (Sensor board) of the DragonBoard using I^{2}C and interrupt lines. A Sony RMTD301 is used as the IR-Transmitter.

RS232 and OTG-to-USB connections are made between the DragonBoard and a host PC. These are used for serial debugging and ADB connectivity, respectively.

A serial log can be seen on a CuteCom graphical serial terminal or any other serial console.

An ADB log can be seen on the terminal by using the command '$adb shell' followed by '#logcat'.

Figure 2. Qualcomm IR-I²C Bridge Demo Setup



Interrupt Configuration

1. Identify the GPIO Number

The DragonBoard makes use of ACCIL_IRQ (Pin6) on the J6 connector for interrupt.

As per the schematics (8060DragonBoard_Schematic_CPU_Carrier_Board_(110311).pdf and 8060DragonBoard_Schematic_Main_Board_(091611).pdf), the GPIO-57 pin of APQ8060 is connected to the ACCEL_IRQ pin.

2. Print the IRQ Number

The interrupt number is printed by adding the following lines to the board support package file found in the following path:

..../8060_v3/kernel/arch/arm/mach-msm/board-msm8x60.c
#define GPIO_IRDETECT_IRQ 57

Under ‘static void __init msm8x60_dragon_init(void)’ add the following lines:

```
uint32_t irq;
gpio_request(GPIO_IRDETECT_IRQ, "IR-DETECT");              // REQUEST GPIO 57
gpio_export(GPIO_IRDETECT_IRQ, 0);                         // EXPORT GPIO 57
gpio_direction_input(GPIO_IRDETECT_IRQ);                   // INPUT DIRECTION GPIO 57
printk(KERN_EMERG " PRINTING IRQ NUMBER \n");              // DEBUG
irq = gpio_to_irq(GPIO_IRDETECT_IRQ);                      // GET IRQ NUMBER
printk(KERN_EMERG " IRQ NUMBER of GPIO-57 : %d\n", irq);   // DEBUG
```

Build the kernel and flash boot.img file onto the DragonBoard as explained in Steps 5 and 7.

The boot log generated during reboot will print the IRQ number as follows:

```
IRQ NUMBER of GPIO-57 : 313
```

Note: Modified ‘board-msm8x60.c’ is given with this document and can be found in the following location:

```
/8060_v3/kernel/arch/arm/mach-msm/board-msm8x60.c
```

3. Identify I²C Device Lines

The I²C device line registered on the J6 connector is identified from the device file of the DragonBoard:

```
…./8060_v3/kernel/arch/arm/mach-msm/device-msm8x60.c
```

It is identified that i2c-5 lines are used on the J6 connector (sensor connector). This device lines are used to read and write from the iCEdragon Demo Board.

4. Interrupt Handling Driver in Kernel

The interrupt handling driver file ‘I2C_irdetect_dev.c’ and respective modified ‘Makefile’ are provided with this document. These files exists in the path below:

```
/8060_v3/kernel/drivers/i2c/
```

Simply copy the above driver file and Makefile to the following Android kernel path:

```
…./8060_v3/kernel/drivers/i2c/
```

Now, proceed to build the kernel as described in the following steps.

5. Build the 8060_V3 Android Kernel for the DragonBoard

a. To clean all the previously-built directories and image files, use the command below:

```
$mke clean
```

b. To build the kernel source code, use these commands:

```
$sudo -s
$cd …./8060_v3/
$./build.sh
```

6. Edit init.rc and Rebuild Kernel

When building the kernel source code for first time after using the '$make clean' command, the default init.rc file will be created in this location:

…./8060_v3/out/target/product/msm8660_surf/root

Note: A modified init.rc file is provided with this document and can be found in the following path:

/8060_v3/out/target/product/msm8660_surf/root/init.rc

Use the commands below to edit the init.rc file:

```
$cd …./8060_v3/out/target/product/msm8660_surf/root
$sudo vi init.rc
```

Add the following lines to the init.rc file:

```
mount debugfs debugfs /sys/kernel/debug
chmod 0777 /dev/i2c-5
chmod 0777 /sys/kernel/debug/IR_detect
chmod 0777 /system/etc/dbus.conf
chown system system /system/etc/dbus.conf
chown system system /dev/i2c-5
chown system system /sys/kernel/debug/IR_detect
```

After editing the init.rc file, save it and rebuild the kernel for the changes to take effect in the boot.img file.

Re-build the kernel source code using the commands below:

```
$sudo -s
$cd …./8060_v3/
$./build.sh
```

Note: Do not use the '$make clean' command during re-build as this will overwrite to default init.rc file.

Once the build is complete, follow the steps below to flash the boot image to the board.

7. Flash the Boot Image to the DragonBoard

a. Run the command below on the terminal to enter Root Mode:

```
$sudo -s
```

b. Reboot the DragonBoard in Fastboot Mode. To do this, hold down button **5** on the keypad during reboot.

The DragonBoard can be rebooted to Fastboot Mode by pressing the **Power** button on the board for few seconds until the power-off option is shown on the touch screen. Now hold down button **5** on keypad and click the **Power Off** option shown on the touch screen. Hold down button **5** until the reboot is complete.

Note: Not holding down button 5 on the keypad during reboot will reboot the board with the existing Android operating system.

c. Verify FastBoot Mode. When the board is in FastBoot Mode mode, the screen is blank. Executing the command below on the terminal with root permission will list the FastBoot device number and its name:

```
$fastboot devices
```

The board is now ready to be flashed with the boot.img file.

d. Type the command below to flash the boot.img file:

```
$cd …./8060_v3/out/target/product/msm8660_surf/
$fastboot flash boot boot.img
```

If the boot.img file is installed correctly, you will see the OKAY and FINISH comments.

e. Re-boot the DragonBoard to see the new boot image in effect. The following command will reboot the board to load the Android operating system:

```
$fastboot reboot
```

8. Build the IR_Scanner Application

a. Import the project to your Eclipse workspace.

The IR_Scanner application source is located in path given with this document:

Application/IR_Scanner

Open the Eclipse IDE with the Android SDK and ADT plug-in installed.

Click **File > Import > General >Existing Project** into the workspace.

Click **Next** and under the Project Options window, browse to select the above project location under the root directory. Click **Next**.

Select the **Android 2.3.3** option and click **Finish**.

The IR_Scanner project is successfully imported into the workspace.

b. Build Java Native Interface files

JNI C files can be found in the following location:

'Application/IR_Scanner/jni'

Open the file **nativeIRDetect.c** from above location and edit line 53 to include the path of the Anrdoid Directory i2c-dev.h file:

```
#include<..../8060_v3/kernel/include/linux/i2c-dev.h>
```

To re-build JNI files, enter the following commands:

```
$cd Application/IR_Scanner/jni
$sudo -s
$ …./android-ndk-r6b/ndk-build
```

Note: Download android-ndk from http://developer.android.com/sdk/ndk/index.html. This will generate the source file 'libnativeIRDetect.so' under the following path:

'Application/IR_Scanner/libs/armeabi/'

c. Clean and build the IR_Scanner project.

Under the Project tab, click the **Clean** option, select the **IR_Scanner** and click **OK**.

To build the IR_Scanner project, select the project IR_Scanner from the Package Explorer window. Under the Project tab, click **Build Project**.

9. Install IR_Scanner Application to Android

The IR_Scanner application can be installed onto the DragonBoard by following the steps below.

a. The DragonBoard is powered up and running Android.

b. Select the IR_Scanner project from the Package Explorer window on the left side of the Eclipse IDE. Click **Run > Run as > Android Application**. This will create and install the IR_Scanner application on the board.

The Android package file (apk) can be found in the following location:

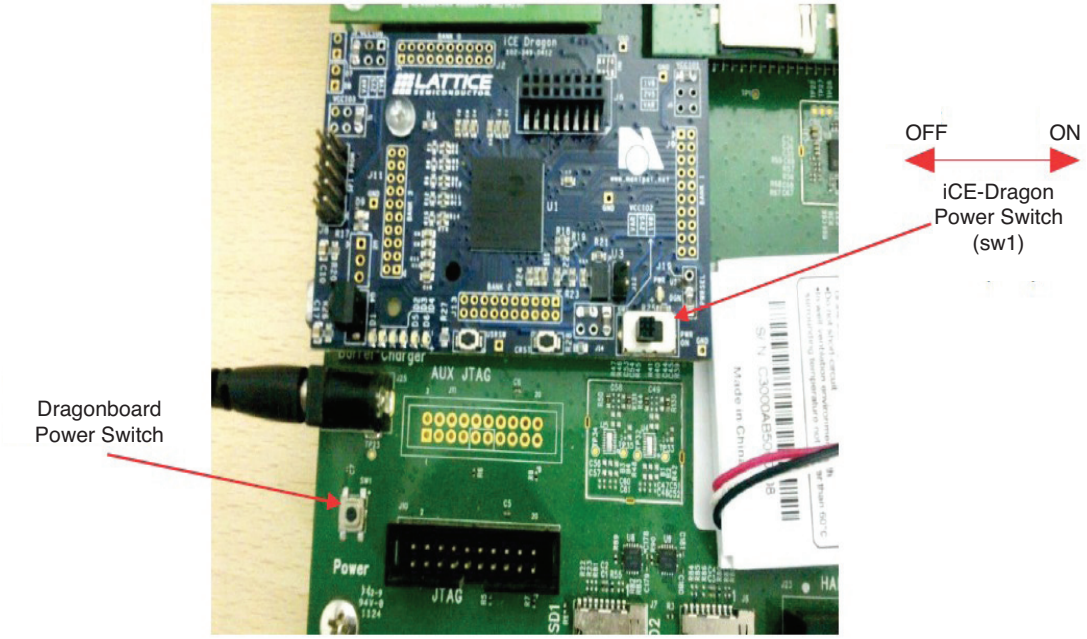
'Application/IR_Scanner/bin'

10. Run the Qualcomm IR-I²C Bridge Demo

a. Power off the iCEdragon Demo Board by toggling the switch (sw1) to the **OFF** state.

b. Power off and Restart the DragonBoard.

c. After the Android boot is complete and the home screen appears, switch on the board by toggling the switch (sw1) to the **ON** position.



d. On the DragonBoard go to the Android application menu and open the application labled **IR_Scanner**. The application logo is shown below.

e. Now the application is ready to detect a key press from the Sony remote (RMTD301).

11. Debugging and Troubleshooting in the Kernel and Application

To debug board support files using printk statements, use printk functions with the KERN_EMERG flag.

Example: Printk(KERN_EMERG "MY PRINT LOG\n");

The printk lines can be shown on a serial console.

To enable the Android Debug Log, add the following line in the Android.mk file of your jni directory:

LOCAL_LDLIBS := -llo

Inside JNI 'C' code use android_log_print lines in the following format to debug:

```
#include<android/log.h>
#define  LOG_TAG    "nativeIrdetect"
```

Example:

```
__android_log_print(ANDROID_LOG_DEBUG, LOG_TAG,"PRINT LOG \n");
__android_log_print(ANDROID_LOG_INFO, "PRINT VALUE=", "%d\n", value);
```

The debug lines can be observed on a serial console by using the '#logcat' command.

If the android application does not respond, follow the steps below in the correct sequence:

a. Power off the iCEdragon Demo Board by toggling the switch (sw1) to the **OFF** position.
b. Close the IR_scanner application process running in the background.
   Android menu > Settings > Applications > Manage Applications > IR_Scanner > Force Stop
c. Power-on the board by toggling the switch (sw1) to the **ON** position.
d. Open the **IR_Scanner** application from the Android menu and the application is ready to detect any key press.

Installing the DragonBoard USB Driver in Windows XP

1. Download and install the Android Software Development Kit (SDK).
2. Start the SDK Manager GUI and install the Google USB Driver package mentioned in the Extras subheading.
3. Copy [Android SDK Root]\extras\google\usb_driver to [Android SDK Root]\msm8655-mdp_usb_driver. If there is no such folder, create a new folder.
4. Open the android_winusb.inf (inside of the newly-created directory) file in Notepad or another non-format appending text editor. Clear the contents of the[Google.NTx86] and [Google.NTamd64] paragraphs so that they contain only the following:

```
[Google.NTx86]
;BSQUARE MDP8655
%SingleAdbInterface% = USB_Install, USB\ VID_05C6&PID_9025&MI_00
%CompositeAdbInterface% = USB_Install, USB\ VID_05C6&PID_9025&MI_00
%SingleBootLoaderInterface% = USB_Install, USB\ VID_05C6&PID_9025&MI_00
(1.0, 31-OCTOBER-2012) Lattice Semiconductor Corporation
8 www.Latticesemi.com
Qualcomm IR-I2C Bridge
[Google.NTamd64]
;BSQUARE MDP8655
%SingleAdbInterface% = USB_Install, USB\ VID_05C6&PID_9025&MI_00
%CompositeAdbInterface% = USB_Install, USB\ VID_05C6&PID_9025&MI_00
%SingleBootLoaderInterface% = USB_Install, USB\ VID_05C6&PID_9025&MI_00
```

5. To find VID number of the connected board, follow these steps:
    a. Go to **My Computer** and right-click **Properties**.
    b. In the System Properties window Go to **Hardware > Device Manager**.
    c. In Device Manager you will find “Android Device”. Expand this and you will find “Android ADB Interface”.
    d. In Android ADB Interface right-click and go to **Properties > Details**. The details tab provides the Device Instance ID.
    e. Copy the VID which is listed between USB\“ ”\.
6. Create and/or open the **C:\Documents and Settings\[yourusername]\.android\adb_usb.ini** file and make sure it contains only the hex string 0x05C6 (obtained from the step above).
7. Power on the device and connect it to the development machine. Be sure to point Windows to your newly-created USB driver directory for your MSM8655. If Windows is unable to find the needed driver(s), point the driver search to the original USB driver located in [Android SDK Root]\extras\google\usb_driver.

Ordering Information

Description	Ordering Part Number	China RoHS Environment-Friendly Use Period (EFUP)
iCEdragon Demo Board	ICE40HX8K-DRAGON-EVN	
Lattice USB Programming Cable	HW-USBN-2A	

Technical Support Assistance

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

Revision History

Date	Version	Change Summary
November 2012	01.0	Initial release.

Appendix A. iCEdragon Demo Board Schematics

Figure 3. Bank 0 Connections

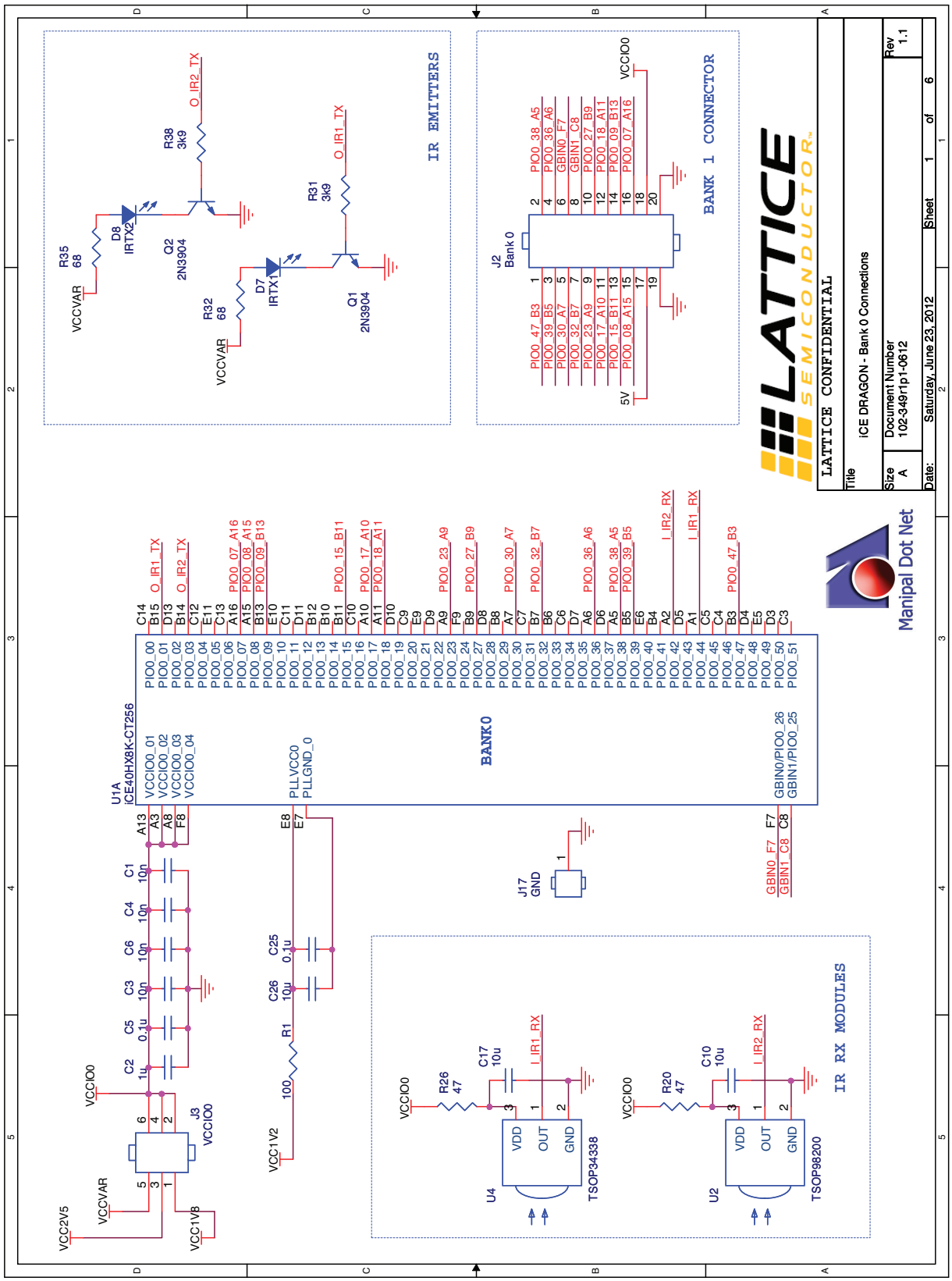


Figure 4. Bank 1 Connections

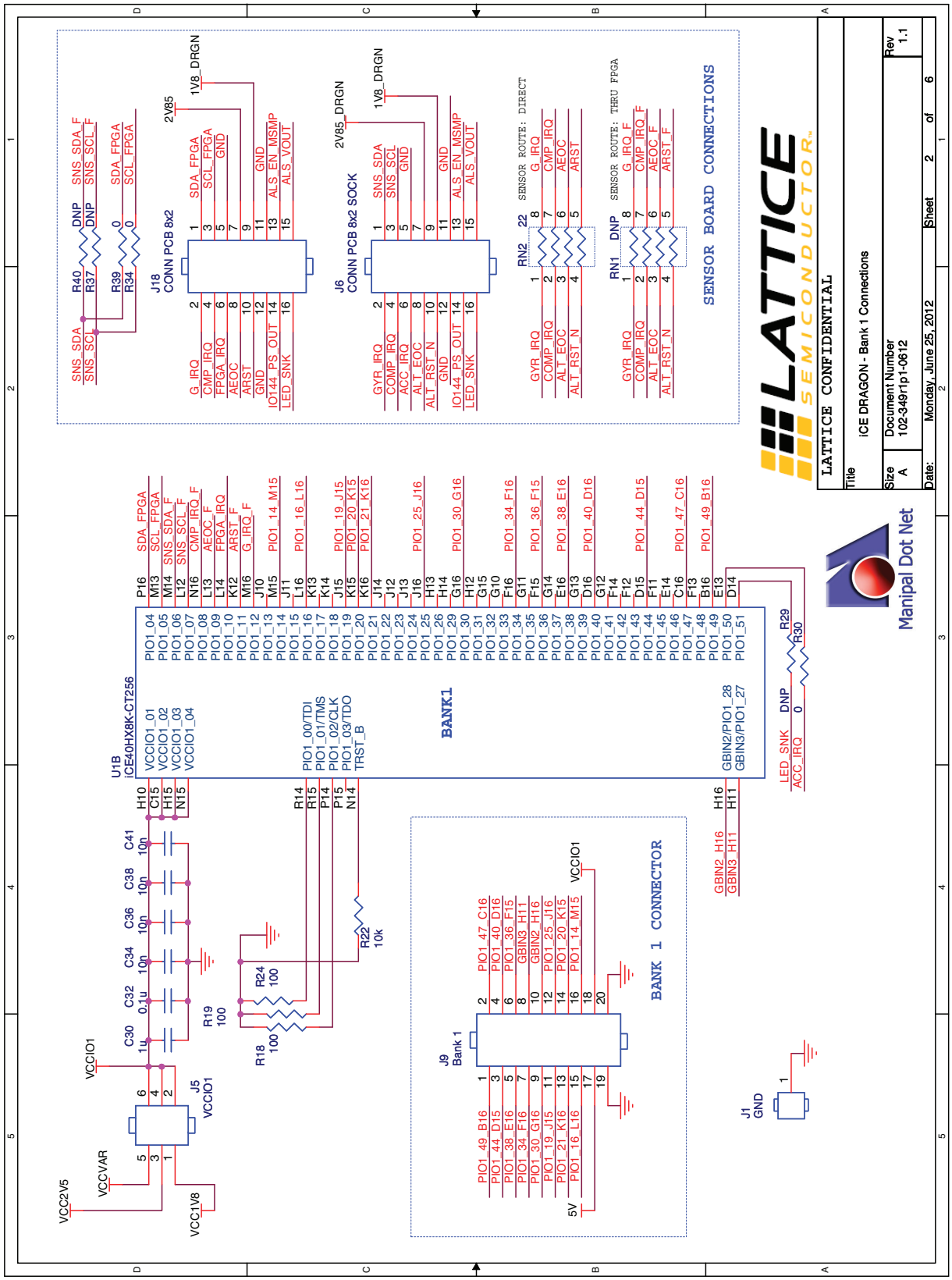


Figure 5. Bank 2 Connections

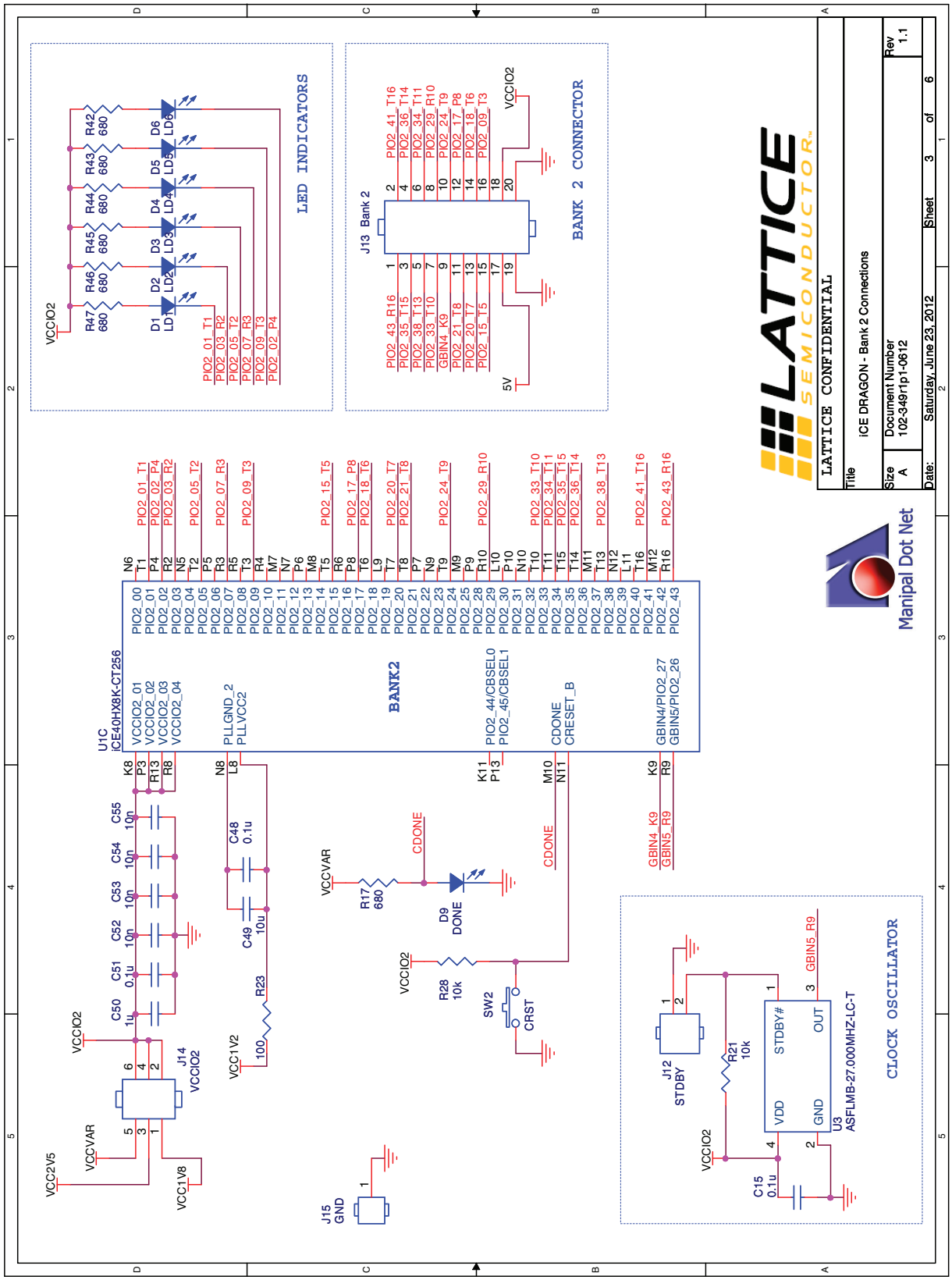


Figure 6. Bank 3 Connections

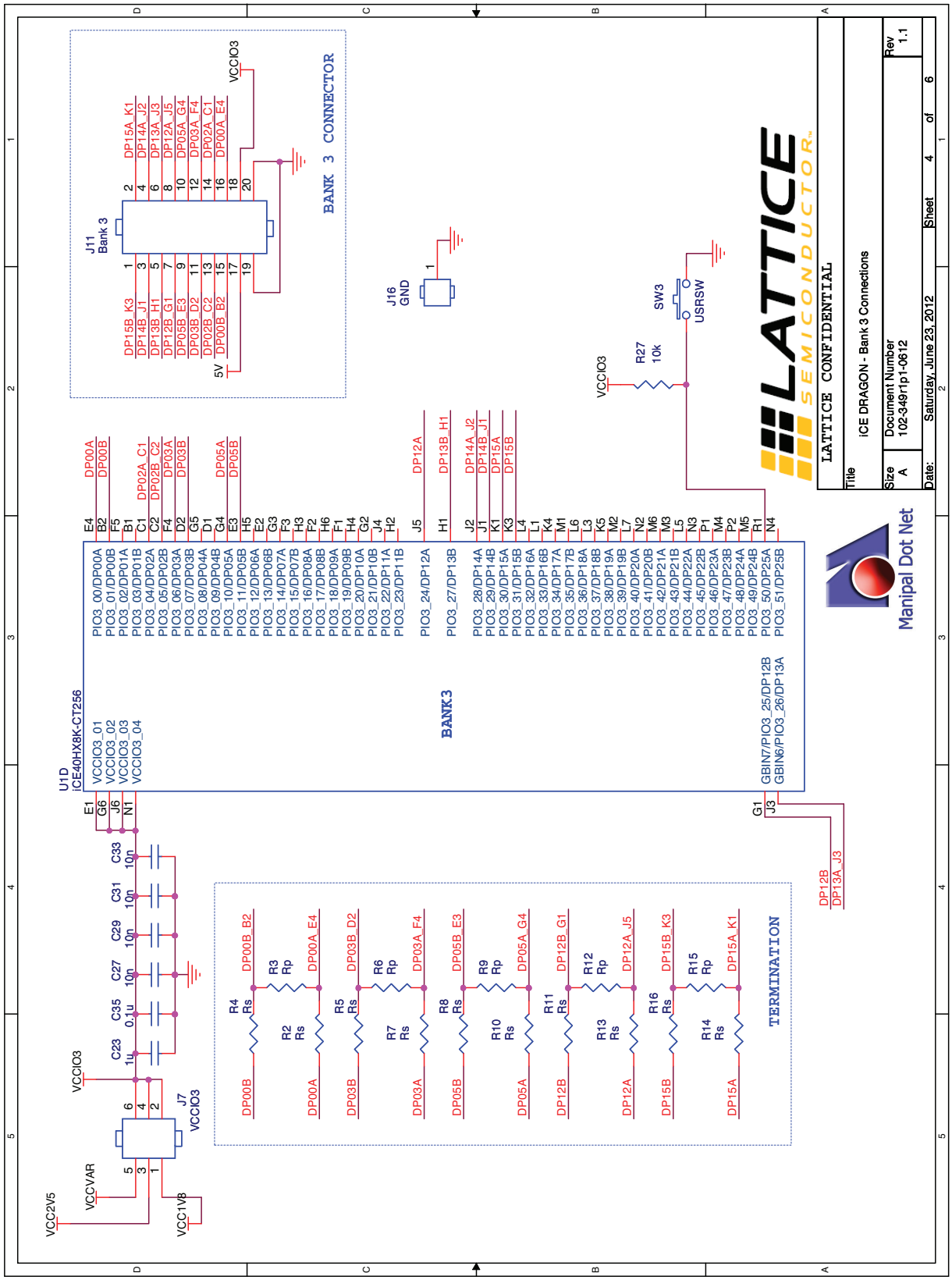


Figure 7. Power Supply

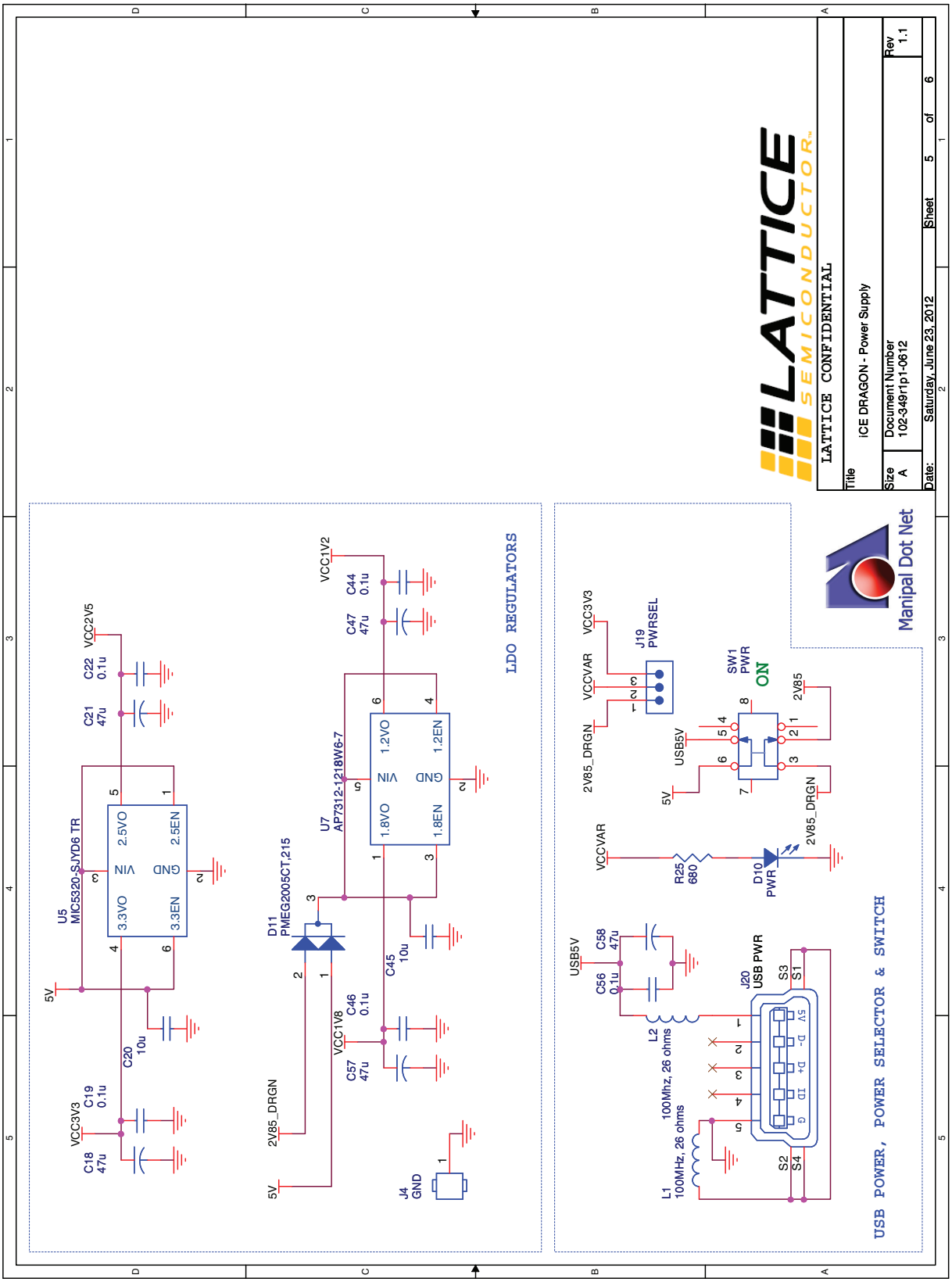


Figure 8. Decoupling and SPI Configuration