UG286: ClockBuilderPro™ Field Programmer Kit

This document describes how to use the Si538x/4x ClockBuilder Field Programmer Kit (“CBPROG-DONGLE”) with ClockBuilder Pro™ (“CBPro”) to support four programming models.

Refer to the text and table below for supported uses:

1. **In-socket Firmware / NVM Programming**
   - Firmware programming of a Si5383/84 device. Silicon Labs provides a 56-pin socket adapter board for this purpose.
   - NVM programming of “base” Si538x/4x devices (e.g., Si5341A-A-GM), or any other factory “pre-programmed” Si538x/4x device (e.g., Si5341A-A12345-GM) which has unused NVM banks. Silicon Labs provides 44-pin and 64-pin QFN socket adapter boards for this purpose.
2. **In-system Firmware / NVM Programming**
   - Firmware programming of a Si5383/84 devices already mounted on a system PCB. Users are encouraged to include a standard 10-pin header on their PCB to allow the Si538x4x Field Programmer board and ribbon cable to easily connect to the USB to SPI/I2C adapter.
   - NVM programming of Si538x/4x devices already mounted on a system PCB. Users are encouraged to include a standard 10-pin header on their PCB to allow the Si538x4x Field Programmer board and ribbon cable to easily connect to the USB to SPI/I2C adapter.
3. **In-system Volatile Register Programming**
   - Devices mounted on a PCB (e.g., use the Design Dashboard and EVB GUIs to inspect status registers, make volatile configuration updates, debug system firmware, etc.).
4. **In-socket Volatile Register Programming**
   - Devices mounted in the socket (e.g., use the Design Dashboard and EVB GUIs to inspect status registers, make volatile configuration updates, debug system firmware, etc.).

KEY POINTS

- Shows and provides a brief explanation of the Field Programmer kit contents
- Points users to CBPro download and installation instructions
- Explains hardware configuration
- Describes the four programming models to use with the CBPROG-DONGLE
- Includes CBPROG-DONGLE and socket board schematics
- Offers bill of materials
- Includes troubleshooting appendix for common issues

Table .1. Supported Programming Models

Location of Target Si538x/4x Device	Software Utility and Programming Model Supported	
	NVM Burn Tool	EVB GUI / Dashboard
In-socket	Yes (1)	Yes (4)
In-system	Yes (2)	Yes (3)

1. Kit Contents

Shown below is a diagram of how the various components in the Field Programmer kit are connected to one of the QFN socket adapter boards, or to a PCB for in-system programming.

Figure 1.1. Example Hardware Configuration (Using QFN Socket Board or Customer PCB)

Figure 1.2 CBPROG-DONGLE Kit Contents on page 2 shows the kit contents for the CBPROG-DONGLE kit. Note in the figure on the following page that the 44-pin, 56-pin, and 64-pin sockets are available separately as part numbers Si538x4x-44SKT-DK, Si538x4x-56SKT-DK, and Si538x4x-64SKT-DK, respectively. The Clock Builder Pro Field Programmer resources including schematics, layout files, and BOM can be found at www.silabs.com/CBProgrammer. Note that the sockets are sold as separate kits.

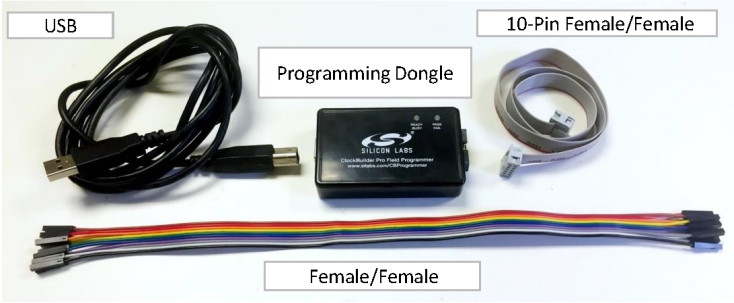

Figure 1.2. CBPROG-DONGLE Kit Contents

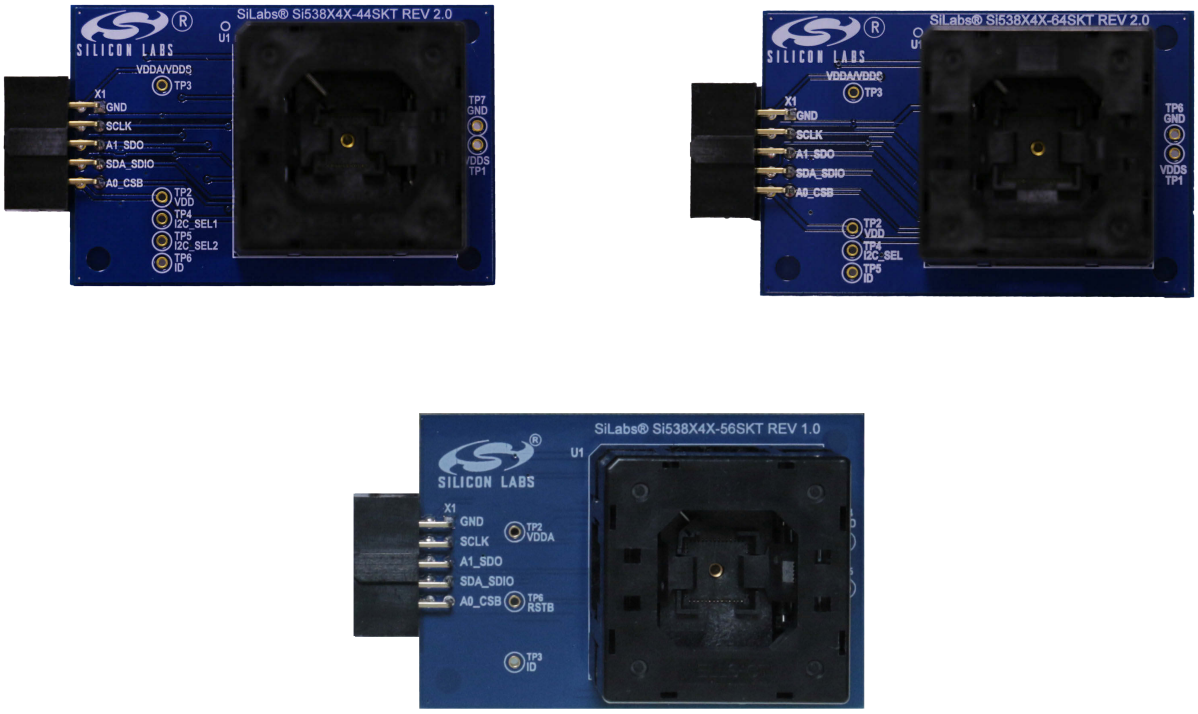

Figure 1.3. Si538x4x-44SKT-DK (left) and Si538x4x-64SKT-DK (right)
Si538x4x-56SKT-DK (bottom) Sockets Sold Separately

2. Software Download and Installation

To install the CBPro software on any Windows 7 (or above) PC, go to http://www.silabs.com/CBPro and download the ZIP file to install the software on your host PC.

3. Hardware Configuration

The Field Programmer Dongle acts as an interface between the CBPro GUI and the target device (any supported Si534x or Si538x IC). Connect the provided USB cable to your PC and the CBPROG-DONGLE. The CBPROG-DONGLE is then connected to the target device using the provided cables or a programming socket, depending upon the four ways you may use the programmer as detailed in Section 4. Ways You can Use the Programmer.

4. Ways You can Use the Programmer

The following four sections describe four ways you can use the CBPROG-DONGLE.

4.1 In-Socket Firmware / NVM Programming

This workflow describes the process of programming of loose devices using the Si538X4X-44SKT, Si538X4X-56SKT or Si538X4X-64SKT programming socket board. For Si538x/4x (not firmware based) devices, this flow will “burn” a complete configuration from CBPro into one of the banks of NVM on the device. Devices shipped from Silicon Labs have two NVM banks available to program (“burn”). For Si5383/84 (firmware based) devices, this flow will flash a complete configuration from CBPro in to the device.

The steps needed to program a device's NVM are as follows:

1. Assuming the CBPro software is installed, connect the CBPROG-DONGLE adapter with the USB cable to the PC on which CBPro was installed. Use the USB extender cable (provided with the kit) if your host PC is located far from the CBPROG-DONGLE.

Figure 4.1. PC to CBPROG-DONGLE Connection

2. Insert a base or previously pre-programmed (e.g. OPN) Si538x/4x device into the socket.

Socket and device Orientation: It is important to ensure the device is in the correct orientation before powering up the board. If not orientated correctly the software has a feature to auto-detect it is not able to read the part. Likely the reason is there is no part in the socket or it is oriented incorrectly. The part will not be damaged if oriented incorrectly. The device has two circles on the part. The smaller circle is the pin 1 indicator. Pin 1 on the socket is lined up with the U1 and dot symbol on the silk screen.

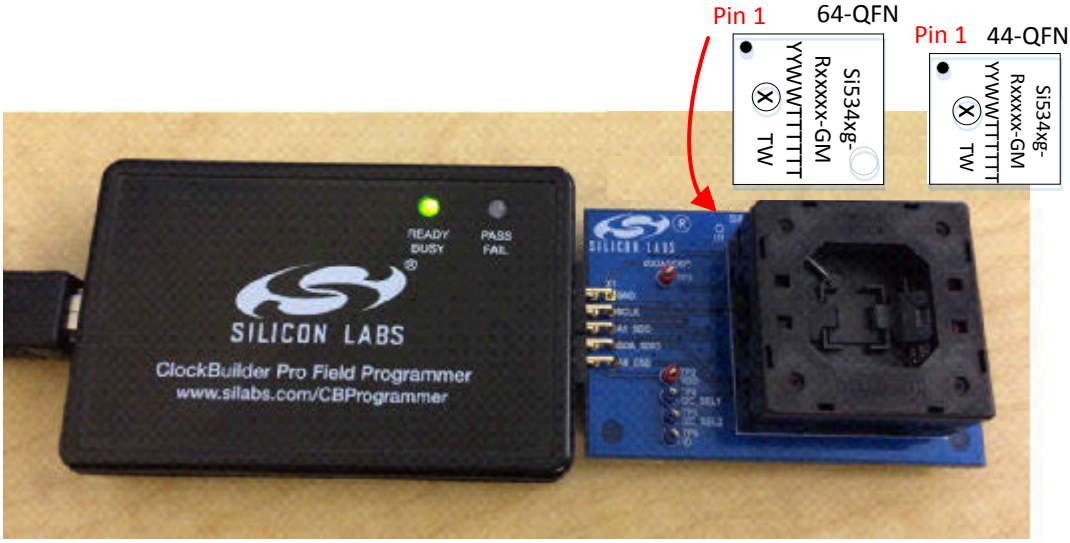

Figure 4.2. Correct Orientation of a Device in the Socket

Note: Power is not applied to the socket's VDD and VDDA pins unless explicit action by you within CBPro. It is safe to:

- Insert or remove a device in the socket before or after the socket has been connected to the main board.
- Insert or remove a device in the socket before or after power has been applied to the main board by connecting the USB cable to your PC.

Power is only applied to the device when you perform a scan or initiate a burn. Power is off at all other times.

3. Connect the QFN Field Programmer Socket Board with the device into the CBPROG-DONGLE.

Figure 4.3. System from PC to Programming CBPROG-DONGLE Board to Field Programmer Socket Board

4. Start ClockBuilder Pro by locating the icon on your desktop or Windows Start Menu.

Figure 4.4. ClockBuilder Pro Icon

5. The ClockBuilder Pro Wizard main menu should now appear, as shown in the figure below. Select the “NVM Burn Tool” as shown. ***Do not select EVB GUI.***

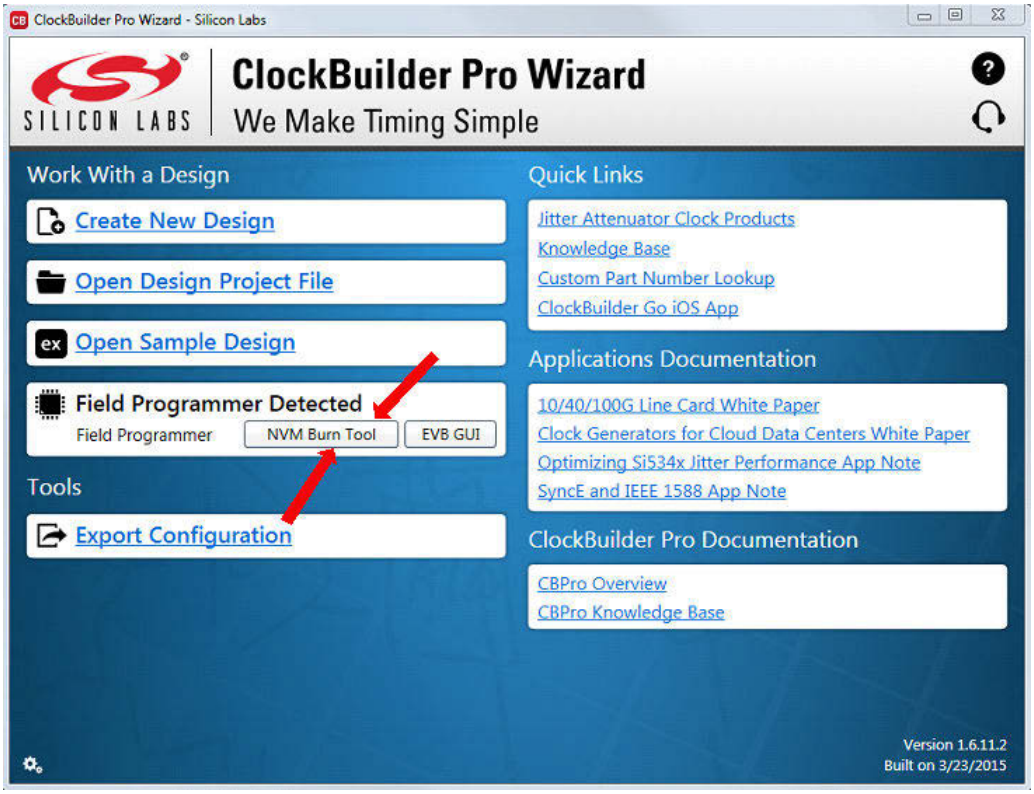

Figure 4.5. ClockBuilder Pro Wizard

6. If this is the first time you are launching the NVM Program Tool and no socket board has been detected, the tool will prompt you to select the device family you are targeting, as shown in the figure below:

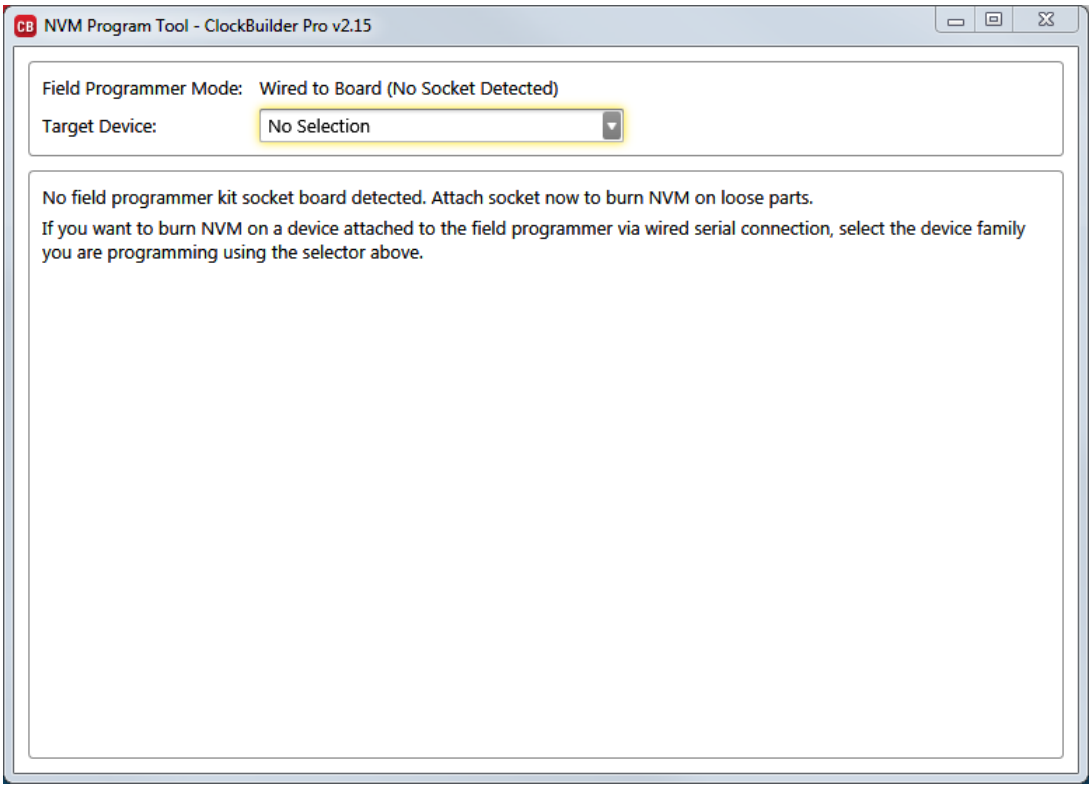

Figure 4.6. Select Device Family Prompt

7. Once you insert the socket in the field programmer, the tool will detect it and automatically load the appropriate programming panel:

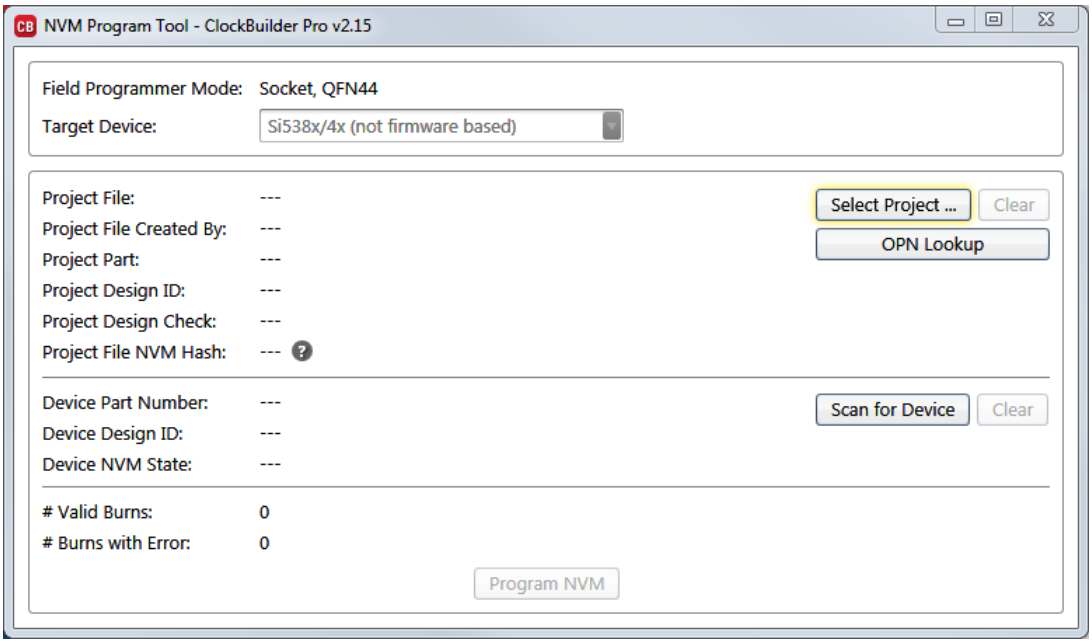

Figure 4.7. Programming Panel

4.1.1 Programming In-socket, Firmware Based Devices

Refer to Figure 4.8 Programming In-socket, Firmware Based Devices on page 10 below.

1. Configure the I2C address and bus speed for the device.
2. Select the firmware source.
   - **Configuration + Program from Project File**
     The configuration defined by the specified project + the firmware release selected in the project file will be used to generate the firmware image that will be flashed on the device. Note that different versions of CBPro may compute configuration registers differently for the same design goals as improvements are made to CBPro.
   - **Configuration + Program from Firmware File**
     Flash a stand-alone hex or binary firmware file to the device. You must have previously exported the file in CBPro, or the file was sent to you by Silicon Labs. The firmware image contains both configuration and program data. This option is useful if you want to ensure the same configuration register data is flashed to the device regardless of the CBPro version this tool is running on. Firmware images can be created from the CBPro dashboard using the Export tool, selecting the stand-alone file option.
3. Click the “Select …” button and select the file to flash to the device.
4. Click the “Scan for Device” button (optional): Click to detect device and report on part number, firmware version, and DESIGN_ID. This is optional. You can click ‘Program NVM' without first scanning and all relevant pre-burn checks will be performed. Note a device scan is also performed after the NVM burn has been completed, regardless of whether the burn completed successfully or not.
5. Click the “Program NVM” button to flash device. In project file mode, CBPro will create a firmware image behind the scenes based on the project file configuration, and then flash this on the device. The firmware download is verified via read back.

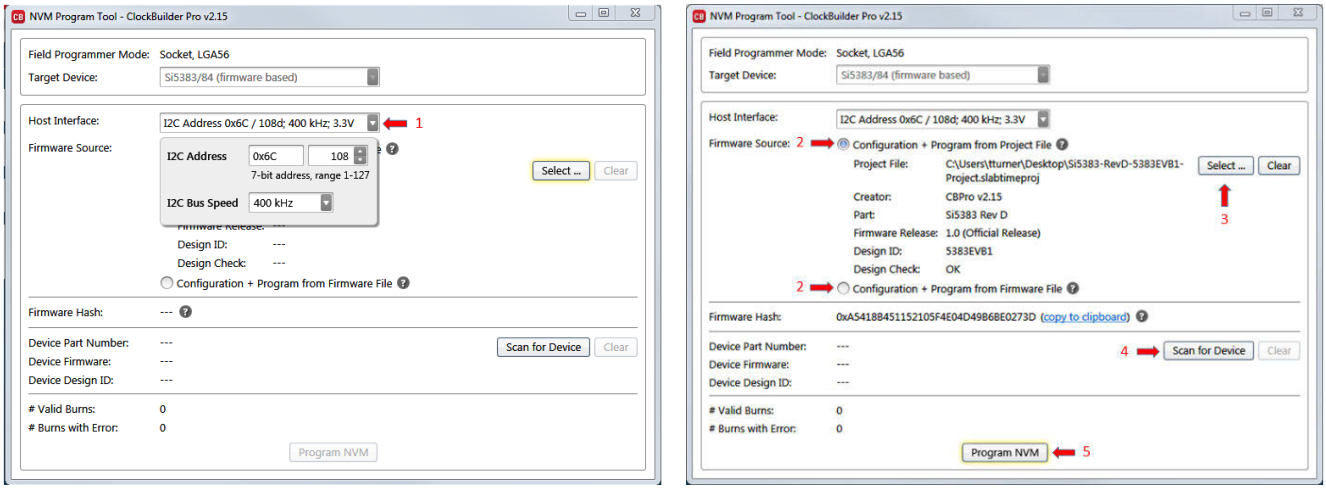

Figure 4.8. Programming In-socket, Firmware Based Devices

4.1.2 Programming In-socket, Non-Firmware Based Devices

Refer to Figure 4.9 Programming In-socket, Non-Firmware Based Devices on page 11 below.

1. Click the “Select Project” button and select the project file.
2. *(Optional)* Click the “Scan for Device” button to detect the device and report on part number, DESIGN_ID, and NVM bank state (number of banks already burned, number available for burn). This is optional. You can click ‘Program NVM' without first scanning and all relevant pre-burn checks will be performed, such as verifying there is a bank available to burn. Note a device scan is also performed after the NVM burn has been completed, regardless of whether the burn completed successfully or not.
3. Click the “Program NVM” button to start the programming flow:
   a. CBPro will compute the registers to program based on the design goals entered in the project file, using the latest algorithms embedded in CBPro.
   b. CBPro will write volatile configuration registers corresponding to the project.
   c. CBPro will initiate a bank burn.
   d. CBPro will force an NVM reload on the device.
   e. CBPro will verify the bank burn by inspecting the bank pointer and read back the programmed registers.
   f. CBPro will rescan for the device and update burn count at the bottom of the window.

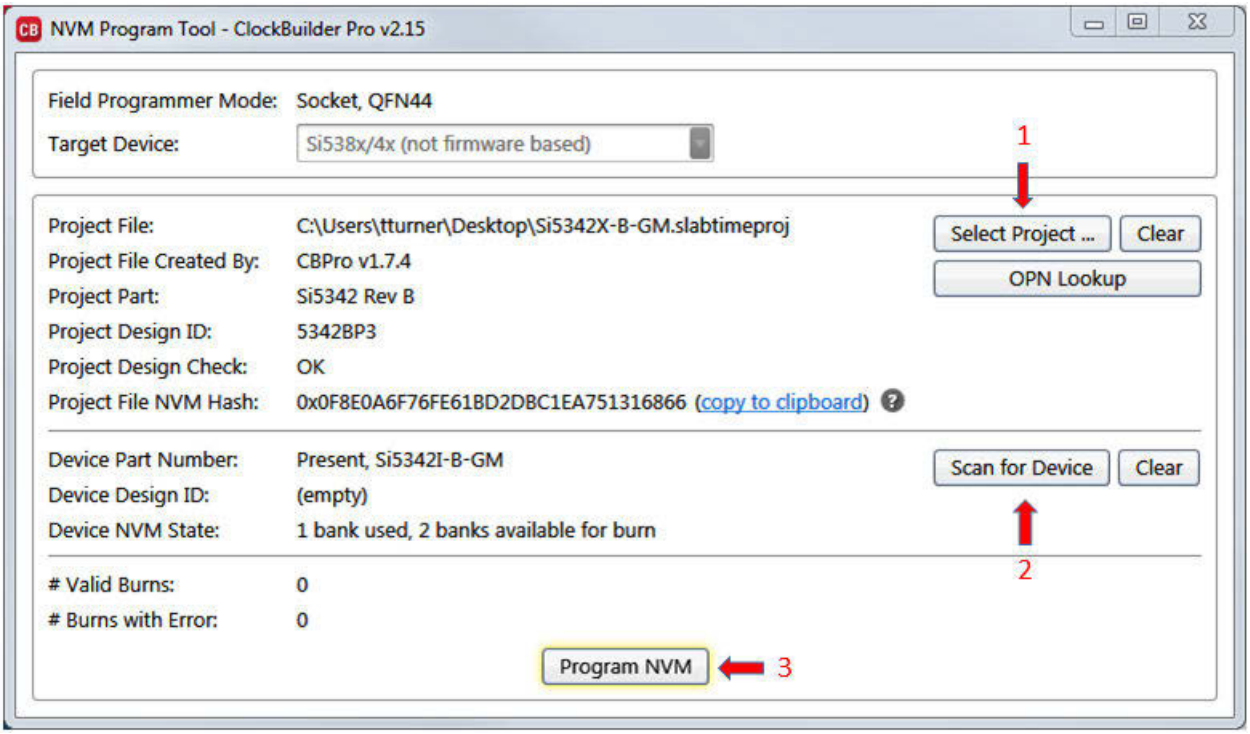

Figure 4.9. Programming In-socket, Non-Firmware Based Devices

4.1.3 In-Socket Programming Status

During the programming process and if the programming is successful, you should see the following windows.

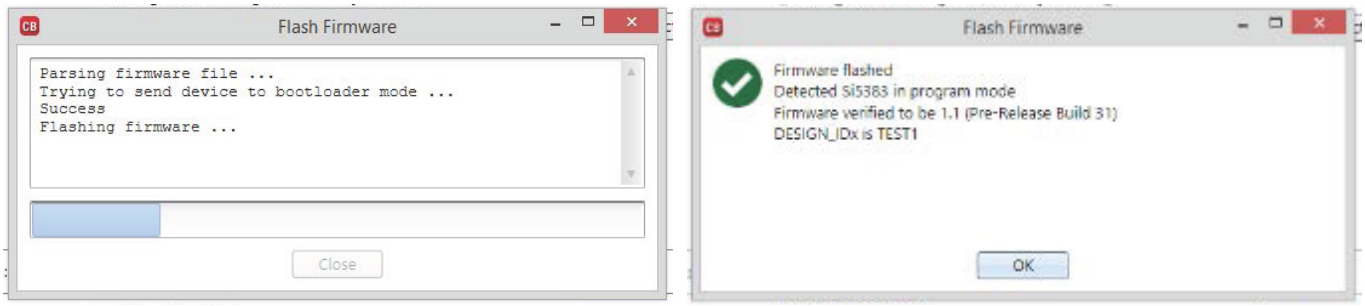

Figure 4.10. In-Socket Programming Status

4.2 In-System Firmware / NVM Programming

This workflow describes the process of programming a device mounted on a PCB. For Si538x/4x (not firmware based) devices, this flow will “burn” a complete configuration from CBPro into one of the banks of NVM on the device, assuming an open NVM bank is available. Devices shipped from Silicon Labs always have two NVM banks available to program (“burn”). If you don’t know how many banks are still open to burn on your target device, CBPro can detect and report the number of remaining NVM banks. For Si5383/84 (firmware based) devices, this flow will flash a complete configuration from CBPro into the device.

The steps needed to program an “in-system” device’s NVM are as follows:

1. Assuming the CBPro software is installed, connect the adapter (CBPROG-DONGLE) board with the USB cable to the PC on which CBPro was installed.

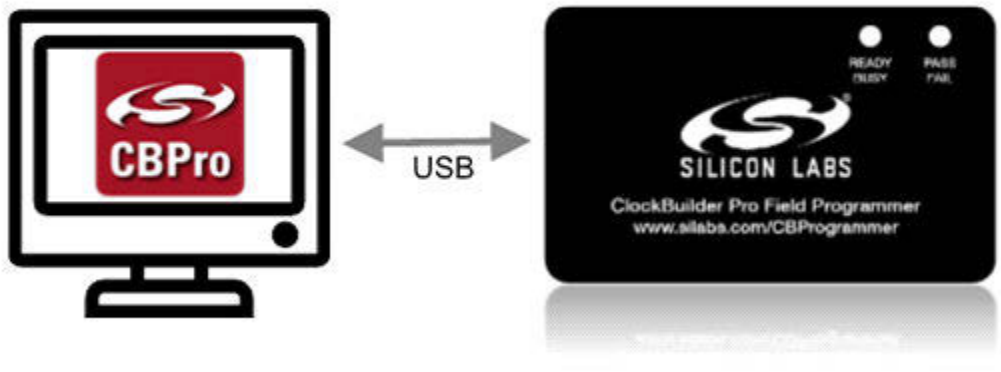

Figure 4.11. PC to CBPROG-DONGLE Connection

2. Lookup and verify the host I/O mode (I2C or SPI), the I2C address, and the interface I/O voltage level compatibility of your host’s I/O voltage (for I2C or SPI) and the device.

   The value set at the device register address of 0x0943 determines how the I/O supply voltages must be configured to communicate reliably with the CBPROG-DONGLE. You can look up your device host I/O voltage using the “OPN Lookup” option in the NVM Burn tool, as shown in Figure 4.12 OPN Lookup Option on page 12.

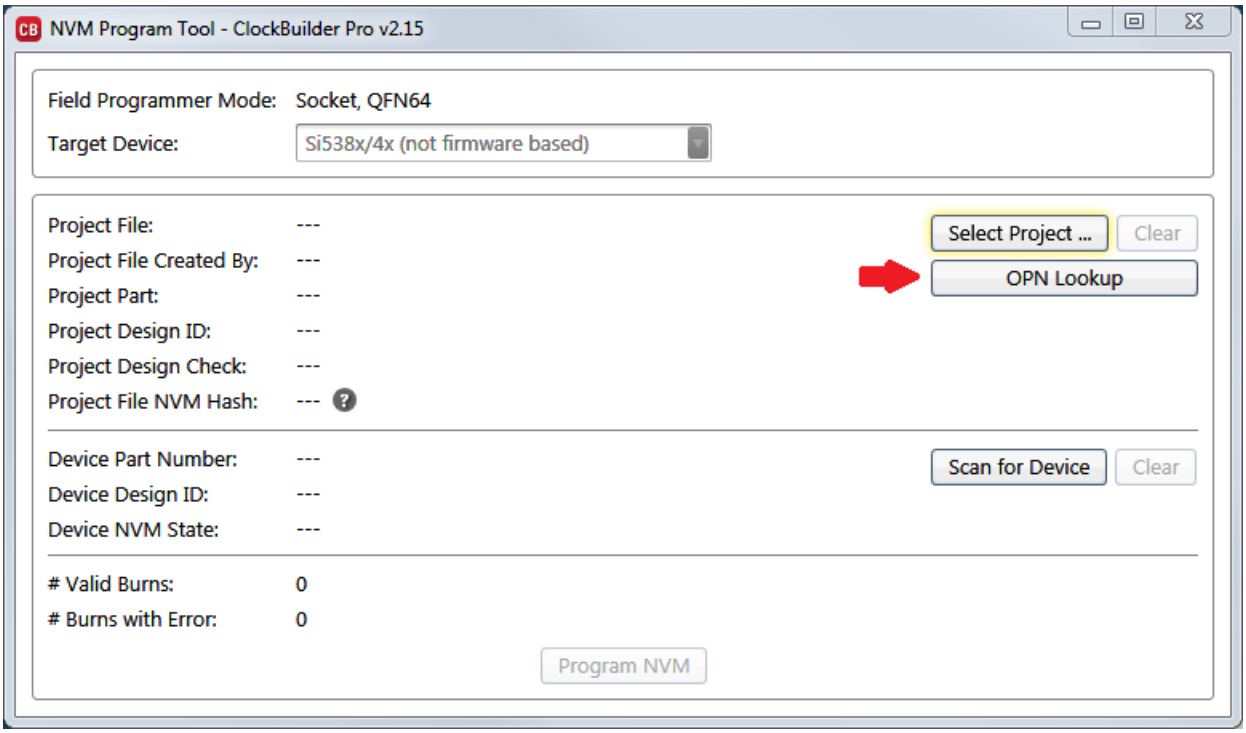

Figure 4.12. OPN Lookup Option

If you have a custom OPN mounted on your board (a part number with a 5 digit code in the middle of the part number, such as Si5346B-A03260-GM), you should look up the host I/O setting (located at address of 0x0943) by selecting the OPN Lookup option. A browser will open and you will then enter in your custom OPN, as shown below.

a. Select “Clock or Buffer”.
b. Enter in your full ordering part number (OPN). E.g., Si5346B-A03260-GM.
c. Click the blue arrow to lookup your OPN to verify the host I/O voltage setting of your device.
d. Click the addendum link.

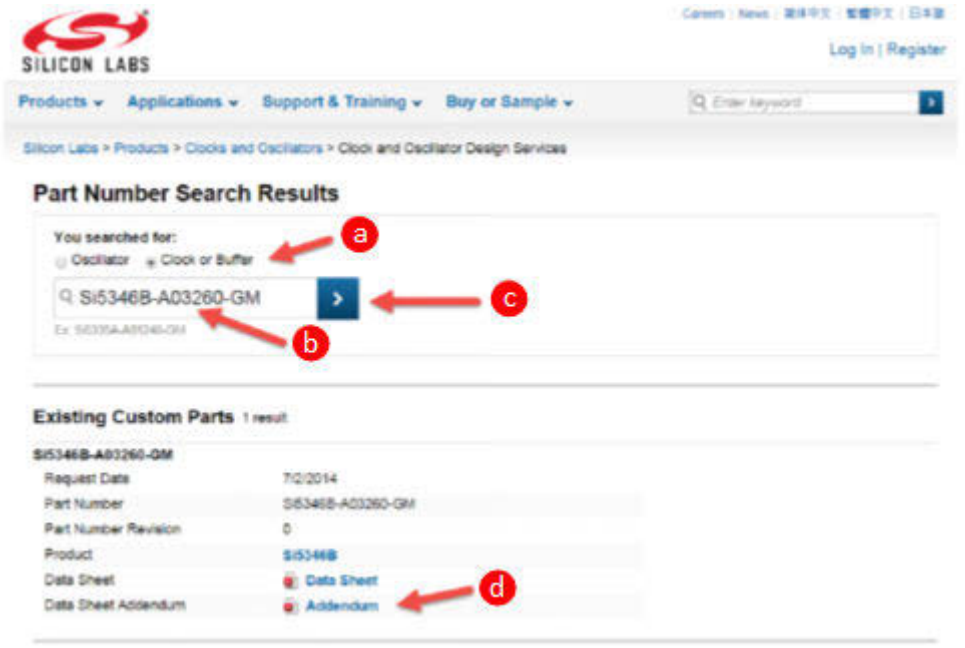

Figure 4.13. OPN Lookup

3. When the utility displays the OPN’s files, click on Addendum to verify the I/O Power Supply setting of your device in the Data Sheet Addendum.
“VDD (Core)” indicates the I/O supply for the I2C/SPI interface will operate from a 1.8 V supply.
“VDDA (3.3 V)” indicates the I/O supply for the I2C/SPI interface will operate from a 3.3 V supply.

Figure 4.14 Finding VDDA Value on page 13 shows an example data sheet addendum showing VDDA (3.3 V).

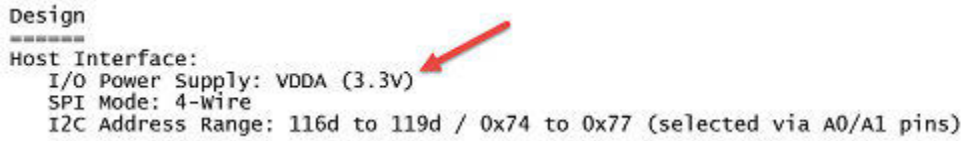

Figure 4.14. Finding VDDA Value

4. Connect/wire the pins of the CBPROG-DONGLE to your host system with the target Si538x4x device. Use the female-to-female ribbon cable to connect to your host board fitted with a standard 10-pin header. This assumes you included the 10-pin header on your PCB and followed the recommended pinout and connections to the target Si438x/4x on your PCB. Note the pinout diagram and descriptions in the table below.

Figure 4.15. Interface Pins on Header (Front View of CBPROG-DONGLE)

Table 4.1. Interface Pin Connections from CBPROG-DONGLE

Pin #	Description	Wire to Your PCB?	I²C	4-wire SPI	3-wire SPI
9	A0_CSB	3- or 4-Wire SPI	Can be used to set I²C address bit A0 high or low. Routed to A0 device pin on the programming Field Programmer Socket Boards.	Drives the chip select signal during SPI transactions	Drives the chip select signal during SPI transactions
10	VDD	Never	Supplies the Core VDD voltage to the device when using a programming Field Programmer Socket Board. Do not use this pin for in-system programming.	Supplies the Core VDD voltage to the device when using a programming Field Programmer Socket Board. Do not use this pin for in-system programming.	Supplies the core VDD voltage to the device when using a programming Field Programmer Socket Board. Do not use this pin for in-system programming.
7	SDA_SDIO	Always	Serial data signal for I²C transactions.	Serial data out to device for 4-wire SPI transactions (MOSI).	Bidirectional Serial data for 3-wire SPI transactions (SDIO).
8	I2C_SEL1	Never	Used to set I2C_SEL signal high to set the device for I²C communication. (Refer to specific part pinout and the programming Field Programmer Socket Board to determine whether to use I2C_SEL1 or I2C_SEL2)	Used to put I2C_SEL signal low for SPI communication. (Refer to specific part pinout and the programming Field Programmer Socket Board to determine whether to use I2C_SEL1 or I2C_SEL2)	Used to put I2C_SEL signal low for SPI communication. (Refer to specific part pinout and the programming Field Programmer Socket Board to determine whether to use I2C_SEL1 or I2C_SEL2)
5	A1_SDO	4-Wire SPI Only	Can be used to set I2C address bit A1 high or low. Routed to A1 device pin on the programming Field Programmer Socket Boards.	Serial data from device for 4-wire SPI transactions (MISO).	Not used
6	I2C_SEL2	Never	Used to set I2C_SEL signal high to set the device for I2C communication. (Refer to specific part pinout and the programming Field Programmer Socket Board to determine whether to use I2C_SEL1 or I2C_SEL2)	Used to put I2C_SEL signal low for SPI communication. (Refer to specific part pinout and the programming Field Programmer Socket Board to determine whether to use I2C_SEL1 or I2C_SEL2)	Used to put I2C_SEL signal low for SPI communication. (Refer to specific part pinout and the programming Field Programmer Socket Board to determine whether to use I2C_SEL1 or I2C_SEL2)
3	SCLK	Always	Serial clock signal for I2C transactions.	Serial clock signal for SPI transactions.	Serial clock signal for SPI transactions.
4	VDDA_VDDS	Never	Supplies the VDDA and VDDS voltages to the device when using a programming Field Programmer Socket Board. Do not use this pin for in-system programming.	Supplies the VDDA and VDDS voltages to the device when using a programming Field Programmer Socket Board. Do not use this pin for in-system programming.	Supplies the VDDA and VDDS voltages to the device when using a programming Field Programmer Socket Board. Do not use this pin for in-system programming.
1	GND	Always	GND	GND	GND

Pin #	Description	Wire to Your PCB?	I²C	4-wire SPI	3-wire SPI
2	ID	Never	The programming Field Programmer Socket Boards provide a voltage on this pin to identify the board. For in-system programming, this pin should be grounded or not connected to any signal.	The programming Field Programmer Socket Boards provide a voltage on this pin to identify the board. For in-system programming, this pin should be grounded or not connected to any signal.	The programming Field Programmer Socket Boards provide a voltage on this pin to identify the board. For in-system programming, this pin should be grounded or not connected to any signal.

4.2.1 I²C Hardware Configuration

For I²C Communication connecting to an external device board, the following pins should be used from the:

CBPROG-DONGLE

- Pin 3: Serial Clock SCLK
- Pin 7: Serial Data SDA
- Pin 1: Ground

Si538x/4x DEVICE

- A0/CS: Drive this pin high or low to set the I²C Address.
- A1/SDO: Drive this pin high or low to set the I²C Address.
- I2C_SEL: Drive this pin high to select I²C communication.

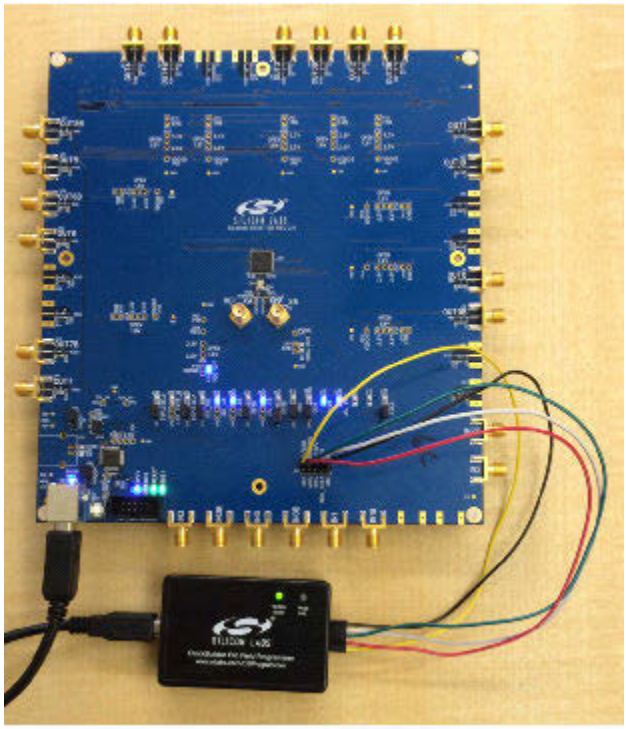

Figure 4.16. Example I2C Connection to External System Target Board Using Jumper Wires (Si5346-EVB)

When using SPI Communication with long wires as shown above it is advisable to use 6 Mb/s bus speed or less.

4.2.2 SPI 3-Wire Hardware Configuration

For 3-wire SPI communication, when connecting to an external device board, the following pins should be used from:

CBPROG-DONGLE
- Pin 3: Serial Clock SCLK
- Pin 7: Serial Data SDIO for Data In and Out
- Pin 9: A0_CSB for Chip Select
- Pin 1: Ground

Si538x/4x DEVICE
- I2C_SEL: Drive this pin low to select SPI communication.

4.2.3 SPI 4-Wire Hardware Configuration

For 4-wire SPI communication, when connecting to an external device board, the following pins should be used from:

CBPROG-DONGLE
- Pin 3: Serial Clock SCLK
- Pin 7: Serial Data SDIO for Data In to device (MOSI)
- Pin 5: A1_SDO for Data Out of device (MISO)
- Pin 9: A0_CSB for Chip Select
- Pin 1: Ground

Si538x/4x DEVICE
- I2C_SEL: Drive this pin low to select SPI communication.

If this is the first time launching the NVM Program Tool, the tool will prompt user to select the device family they are targeting:

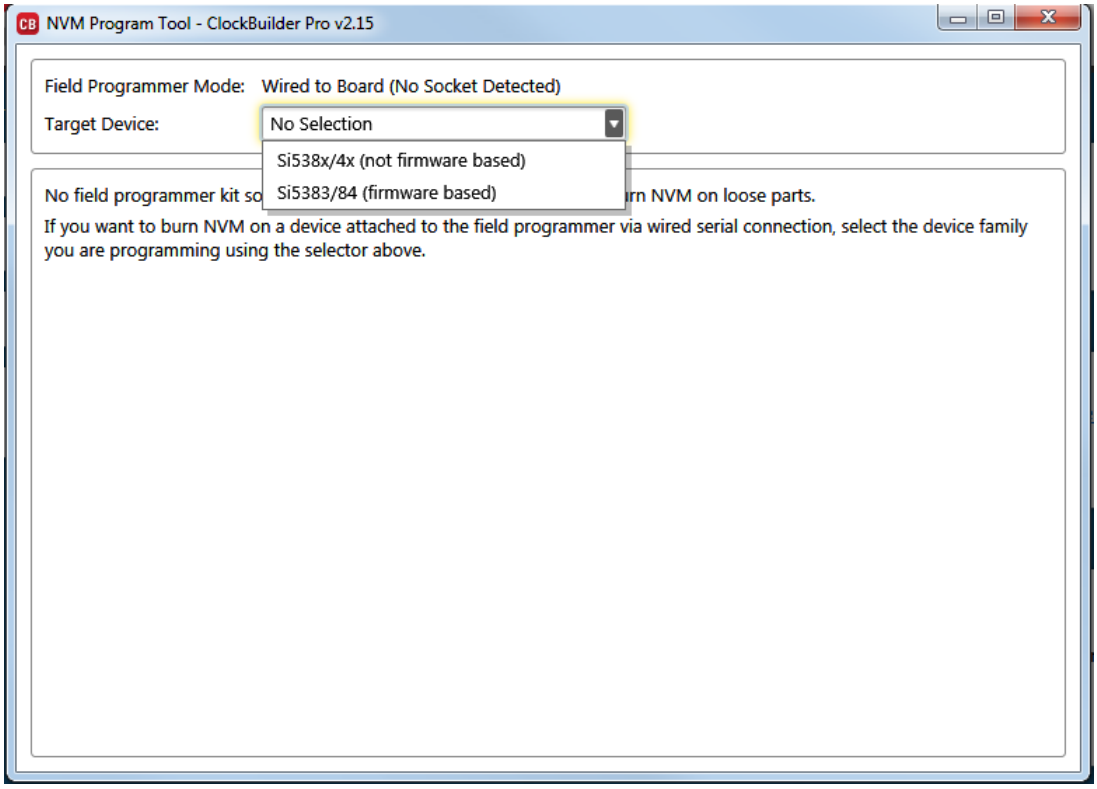

Figure 4.17. NVM Program Tool, Select Device Family

4.2.4 Programming In-system, Firmware Based Devices

Refer to Figure 4.18 Programming In-system, Firmware Based Devices on page 17 below.

After verifying the CBPro Dongle to device connections, execute the following steps. This example assumes a device is configured with an I2C address of 0x6F, and an I²C bus speed of 400 kHz.

1. Select “Si5383/43 (firmware based)” in the Target Device drop down.
2. Click the Host Interface drop down:
   a. Enter the I²C address of the device.
   b. Select the communication bus speed.
3. Select the firmware source.
   - Configuration + Program from Project File
     The configuration defined by the specified project + the firmware release selected in the project file will be used to generate the firmware image that will be flashed on the device. Note that different versions of CBPro may compute configuration registers differently for the same design goals as improvements are made to CBPro.
   - Configuration + Program from Firmware File
     Flash a stand-alone hex or binary firmware file to the device. You must have previously exported the file in CBPro, or the file was sent to you by Silicon Labs. The firmware image contains both configuration and program data. This option is useful if you want to ensure the same configuration register data is flashed to the device regardless of the CBPro version this tool is running on. Firmware images can be created from the CBPro dashboard using the Export tool, selecting the stand-alone file option.
4. Click the “Select Project …” button and select the project file to be written to the device.
5. *(Optional)* Click the “Scan for Device” button to detect device and report on part number, firmware version, and DESIGN_ID. This is optional. You can click Program NVM' without first scanning and all relevant pre-program checks will be performed. Note a device scan is also performed after the NVM programming has been completed, regardless of whether the programming completed successfully or not.
6. Click the “Program NVM” button to flash device. In project file mode, CBPro will create a firmware image behind the scenes based on the project file configuration, and then flash this on the device. The firmware download is verified via read back.

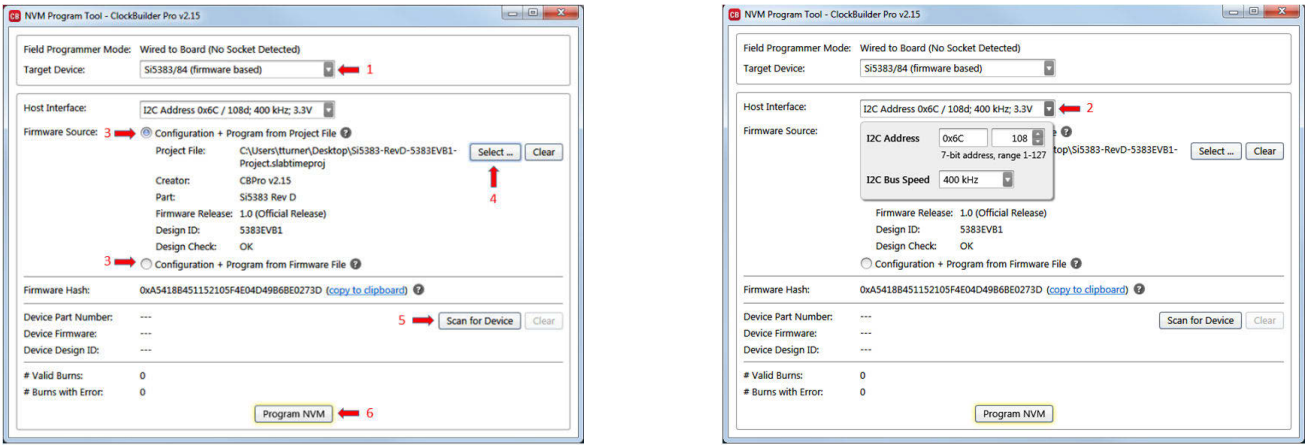

Figure 4.18. Programming In-system, Firmware Based Devices

4.2.5 Programming In-system, Non-firmware Based Devices

Refer to Figure 4.19 Programming In-system, Non-firmware Based Devices on page 18 below.

After verifying the CBPro Dongle to device connections, execute the following steps. This example assumes a device is configured with the host I²C interface operating in 3.3 V I/O mode with an I²C address of 0x6F, and an I²C bus speed of 400 kHz.

1. Select “Si538x/4x (not firmware based) in the Target Device drop down.
2. Click the Host Interface drop down: (Review: **host I/O mode (I2C or SPI)**, **the I2C address**, and **I/O voltage level** to determine these settings)
   a. Select communication protocol for the device.
   b. Select the I/O voltage for the device
   c. For I²C, enter the address of the device.
   d. Select the communication bus speed.
3. Click the “Select Project …” button and select the project file to be written to the device.
4. (Optional) Click the “Scan for Device” button to detect the device and report on part number, DESIGN_ID, and NVM bank state (number of banks already burned, number available for burn). This is optional. You can click Program NVM' without first scanning and all relevant pre-programming checks will be performed, such as verifying there is a bank available to burn. Note a device scan is also performed after the NVM burn has been completed, regardless of whether the burn completed successfully or not.
5. Click the “Program NVM” button to start the programming flow:
   a. CBPro will compute the registers to program based on the design goals entered in the project file, using the latest algorithms embedded in CBPro.
   b. CBPro will write volatile configuration registers corresponding to the project.
   c. CBPro will initiate a bank burn.
   d. CBPro will force an NVM reload on the device.
   e. CBPro will verify the bank burn by inspecting the bank pointer and read back the programmed registers.
   f. CBPro will rescan for the device and update burn count at the bottom of the window.

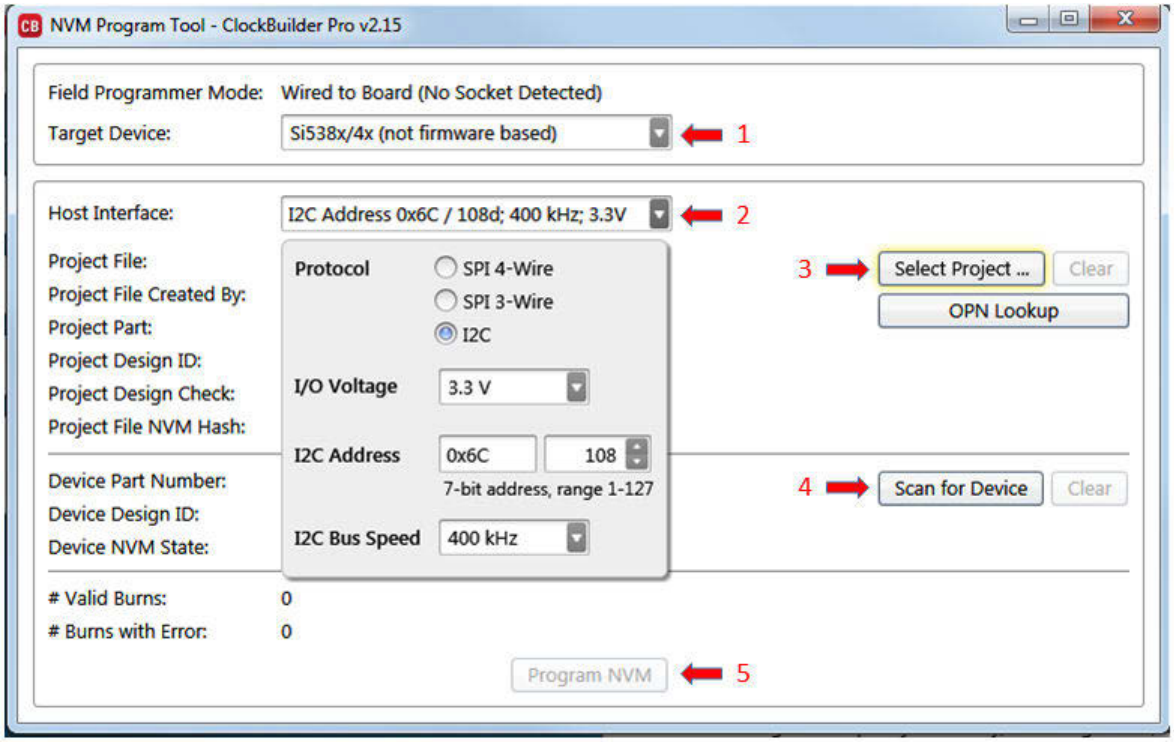

Figure 4.19. Programming In-system, Non-firmware Based Devices

4.2.6 Programming Status

During the programming process and if the programming is successful, you should see the following windows:

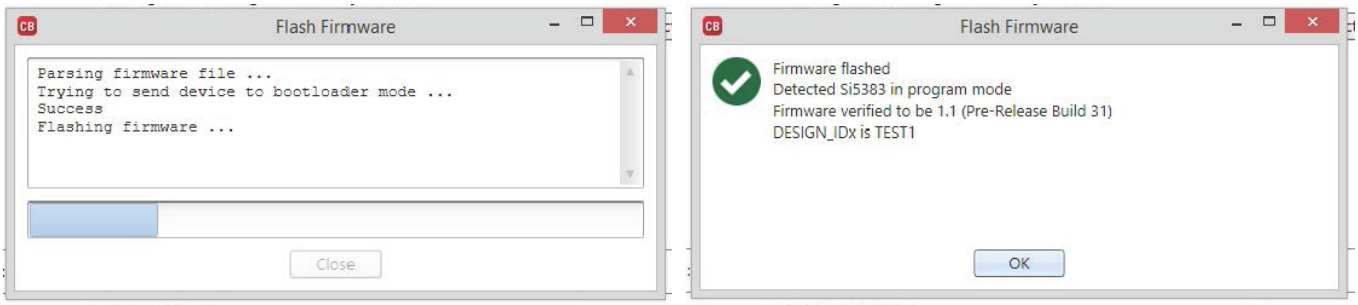

Figure 4.20. Programming Status

4.3 In-System Volatile Register Programming and Register Debug

This workflow allows users to use the full CBPro configuration Wizard and EVB GUI to make volatile changes to a device's configuration and inspect the state of various status registers. There are two ways you can interact with your PCB-based device using the field programmer:

- Use CBPro Design Dashboard to edit your device configuration, and write out changes directly to your device.
- Launch the EVB GUI, to inspect registers.

All of the relevant CBPro features available when working with a Silicon Labs EVB will be available to you, with these exceptions:

- There is no voltage regulator control or voltage/current readings of any kind.
- You must configure the host interface settings so that CBPro can use the device correct communication scheme/wire out.
- If you write out your design/project file, all registers configured via the "Host Interface" section of the wizard **are** written to the device (these registers are skipped when writing a design to a Silicon Labs EVB).

4.3.1 Using the CBPro Design Dashboard

When you launch CBPro, instead of clicking the NVM Burn Tool, open your existing project file or a sample file to open the design dashboard window as shown in the figure below.

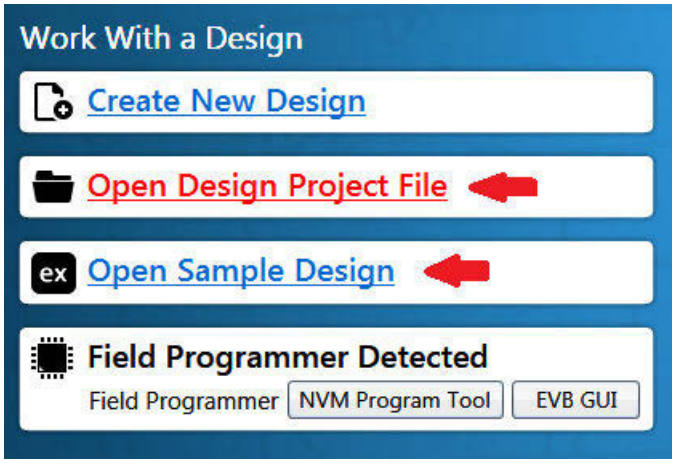

Figure 4.21. Open Design Project File, and see Field Programmer Detected

4.3.2 Overview of CBPro Configuration Wizard and the Field Programmer

When you open a ClockBuilder Pro project file, you are taken to the design dashboard. This is a gateway to perform activities against your design, including writing your project's configuration to a device using the CBPro Dongle. For example, in the figure below, a Si5345 project has been opened and the CBPro Dongle has been detected, and no socket is present:

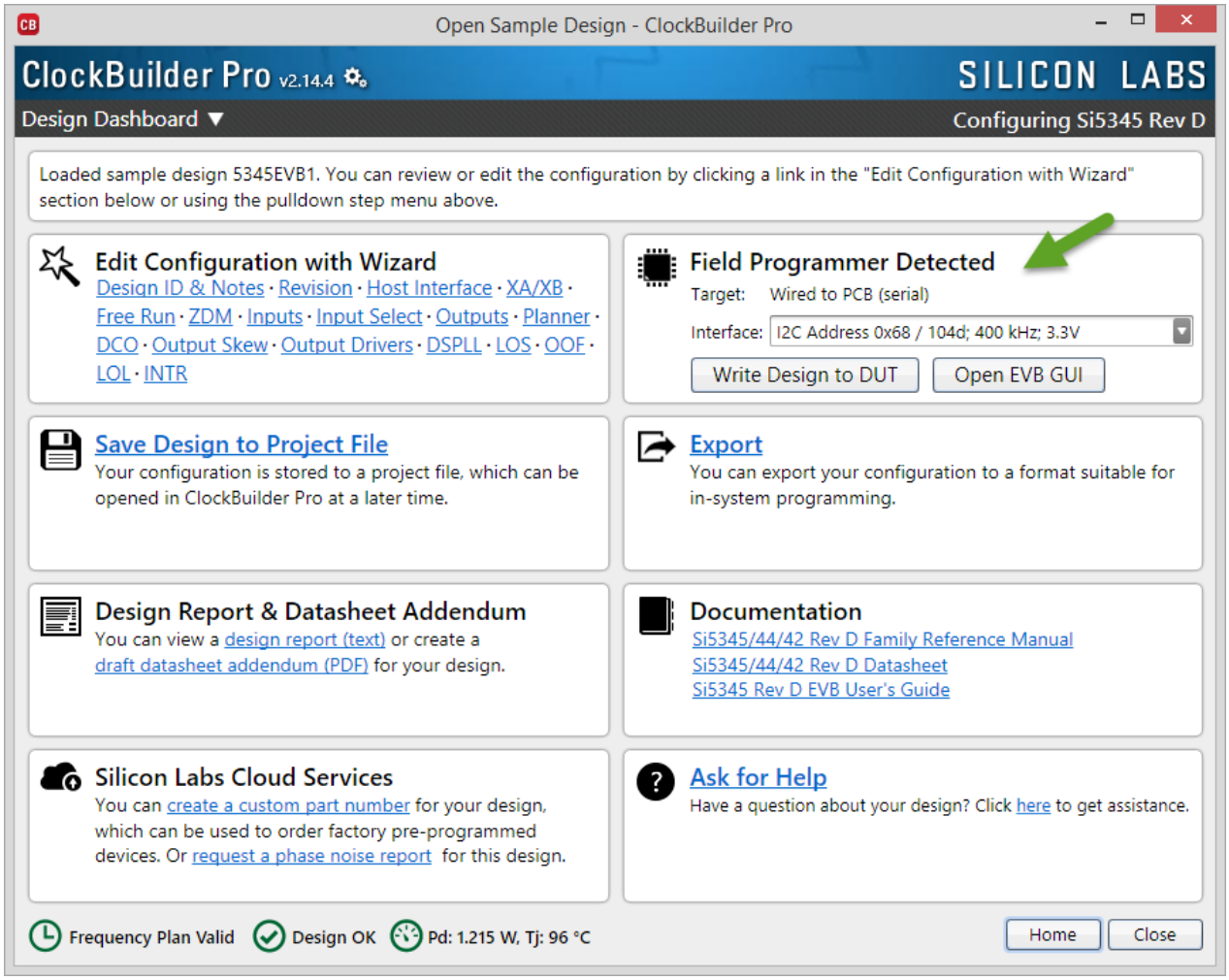

Figure 4.22. Overview of CBPro Configuration Wizard and the Field Programmer

With a click of the "Write Design to DUT" button, you can reconfigure the Si5345 in-system to test changes to your design. The "Open EVB GUI" button can be used to launch the EVB GUI and peek/poke registers on the in-system device. See Section 4.3.4 Using the EVB GUI with In-system Devices to learn more.

4.3.2.1 Using the Dashboard with In-system Devices

If the CBPro Dongle is connected via USB and detected by CBPro, you will see will see a pulldown to configure the host interface between the dongle and your PCB, as shown in the figure below. Refer to Section 4.2 In-System Firmware / NVM Programming for information to connect the CBPro Dongle to your hardware.

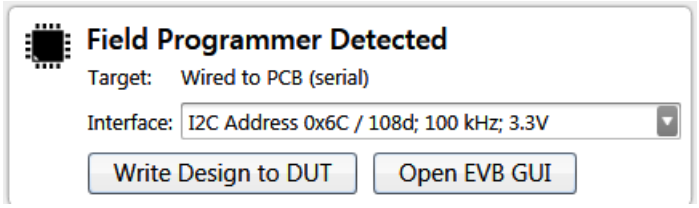

Figure 4.23. Field Programmer Detected

Click the interface pulldown to configure the communication interface, as shown in the figure below. For firmware based devices (e.g. Si5383), the I2C address and bus speed need to be configured. For non-firmware based devices (e.g. Si5340, Si5341), The communication protocol and the I/O voltage need to be configured. If the communication protocol is I2C, the address and bus speed will need to be configured as well.

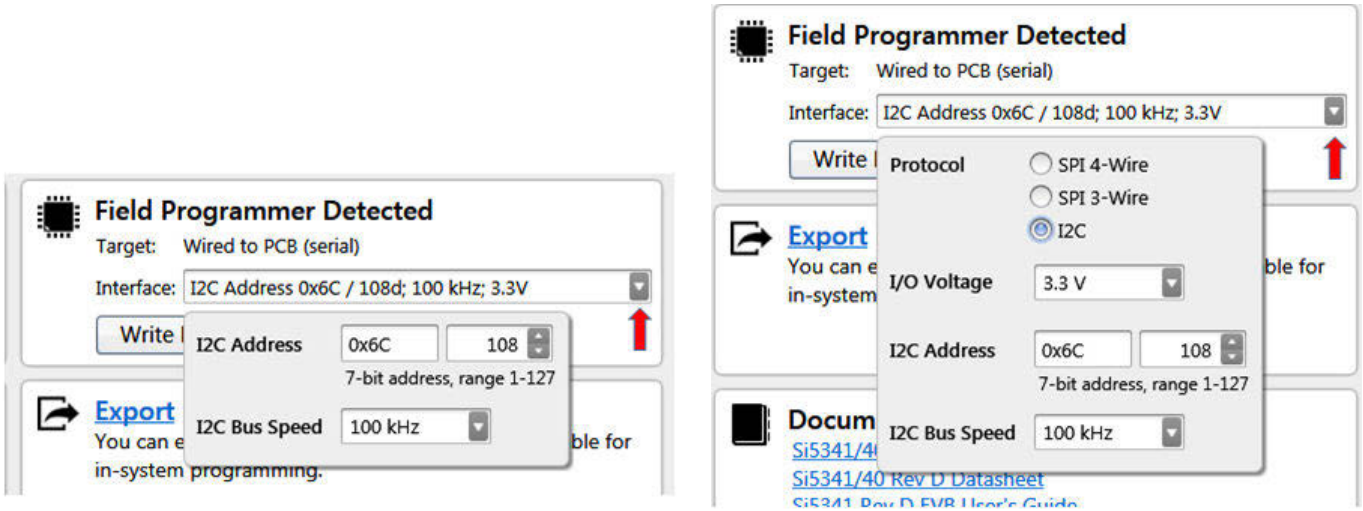

Figure 4.24. Communication Interface Selection

Once configured, you can write out your design to the device by clicking the Write Design to DUT button:

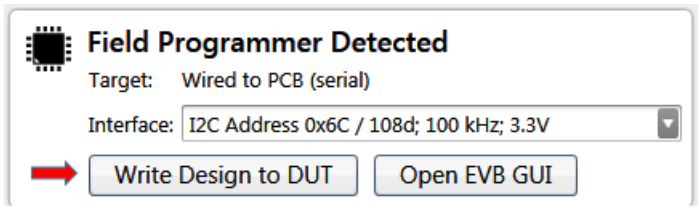

Figure 4.25. Write Design to DUT

Or on any configuration page in the wizard:

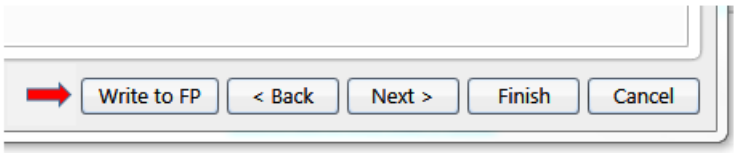

Figure 4.26. Write to FP

When you initiate a project write to the DUT, CBPro will first try to verify the DUT is present via the communication interface you have configured. This is normally accomplished by trying to read device identification register on the device, such as PN_BASE on Si538x/4x devices.

If it cannot read these registers, the DUT write will be aborted and you will see an error message like the example shown in the figure below:

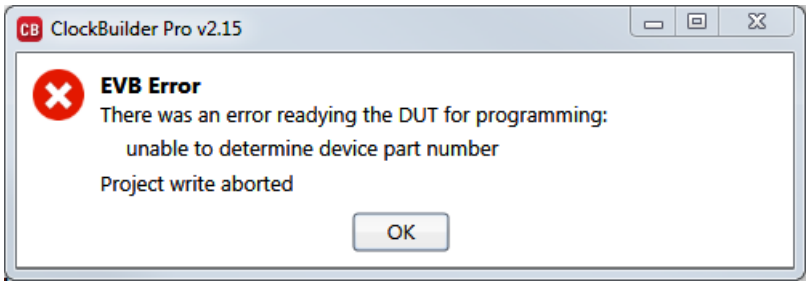

Figure 4.27. Error Message

4.3.2.2 Using the CBPro Dashboard with In-socket Devices

In the design dashboard, you will see a pulldown to configure the host interface between the CBPro Dongle and the socket. If the connected socket is not compatible with the selected CBPro project file, an error message will be displayed and the interface configuration pulldown will be disabled, as shown in the figure below.

Figure 4.28. Socket Compatibility

Click the interface pulldown, configure the interface, and click the slider power to turn on the socket power. For firmware based devices (e.g. Si5383), the I2C address and bus speed need to be configured. For non-firmware based devices (e.g. Si5340, Si5341), the communication protocol and the I/O voltage need to be configured. If the communication protocol is I2C, the address and bus speed will need to be configured, as shown in the figure below.

Note: Manually powering up the socket is an optional step. If you click the "Write Design to DUT" button, CBPro will automatically power up the socket (and you will see it switch from Off to the On state). Socket power refers to VDD and VDDA power on the device.

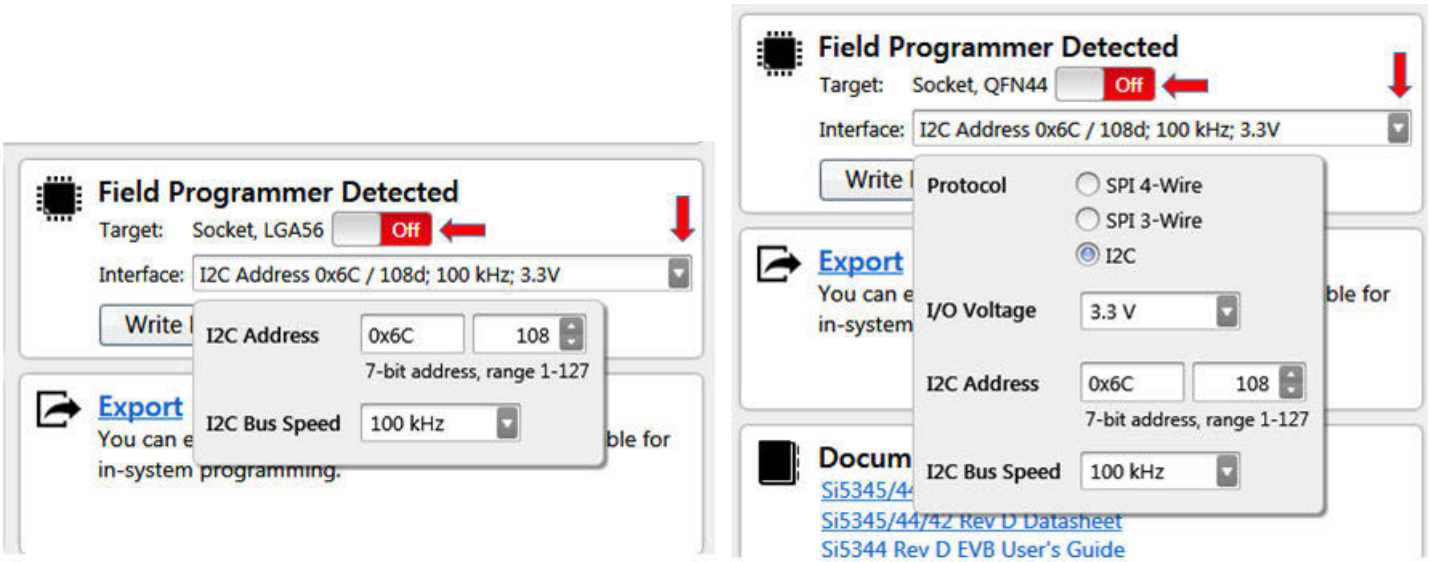

Figure 4.29. Interface Settings

Once configured, you can write out your design to the device by clicking the Write Design to DUT button:

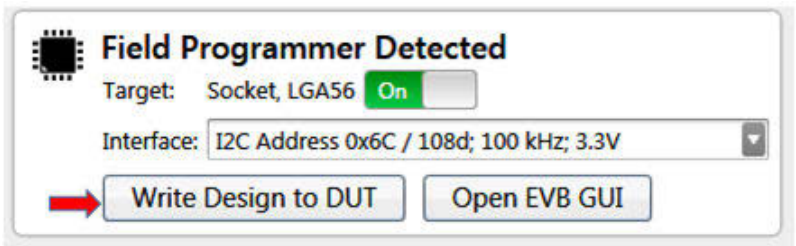

Figure 4.30. Write Design to DUT

Or on any configuration page in the wizard:

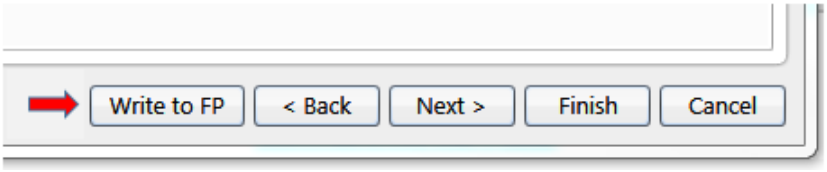

Figure 4.31. Write Design to FP

4.3.3 Launching the CBPro EVB GUI

From the CBPro Wizard screen, click the EVB GUI button to open the EVB GUI screen.

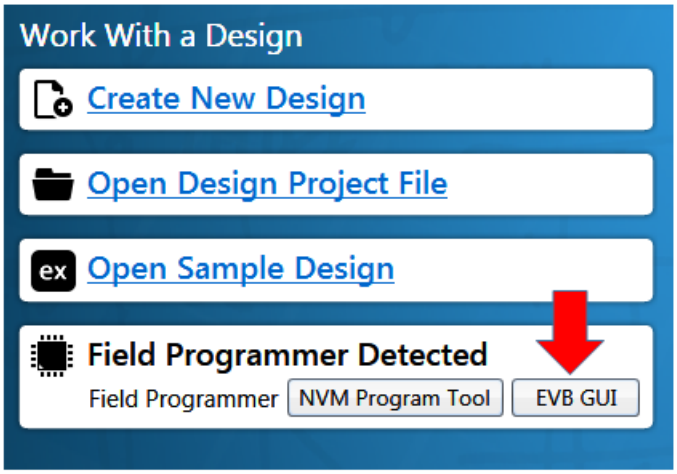

Figure 4.32. Open EVB GUI Screen

If this is the first time launching the EVB GUI and no socket board is detected, the tool will prompt user to select the device family they are targeting:

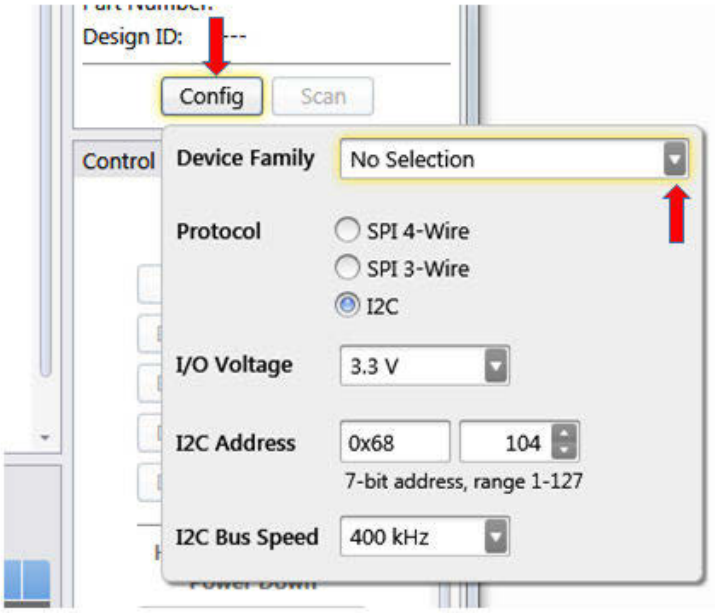

Figure 4.33. Select Device Family Prompt

If a socket is connected, the family is auto selected based on the socket. The tool polls for socket state every 500 milliseconds and will detect if a socket is present or has been changed.

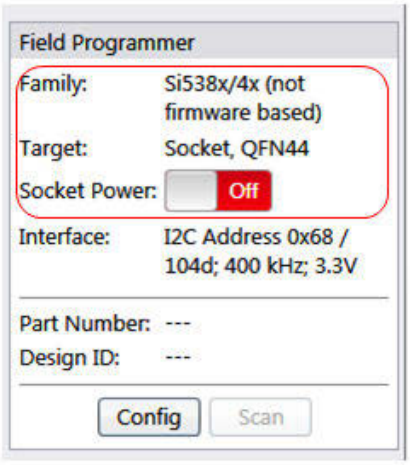

Figure 4.34. Socket Detected, Auto-selected Family Prompt

4.3.4 Using the EVB GUI with In-system Devices

Connect the CBPro Dongle to the PCB mounted device. Refer to Section 4.2 In-System Firmware / NVM Programming for information to connect the CBPro Dongle to your hardware. Click the Config button and click the Device Family pulldown to select either a firmware based device or a non-firmware based device. Then configure the communication protocol, bus speed and I/O voltage (non-firmware devices) for the device, as shown in the figure below.

Note: For firmware based devices the communication protocol available is I2C with a 3.3 V I/O voltage. For non-firmware based devices, there is a selection of SPI 4-wire, SPI 3-wire, or I2C and the I/O voltage must be selected.

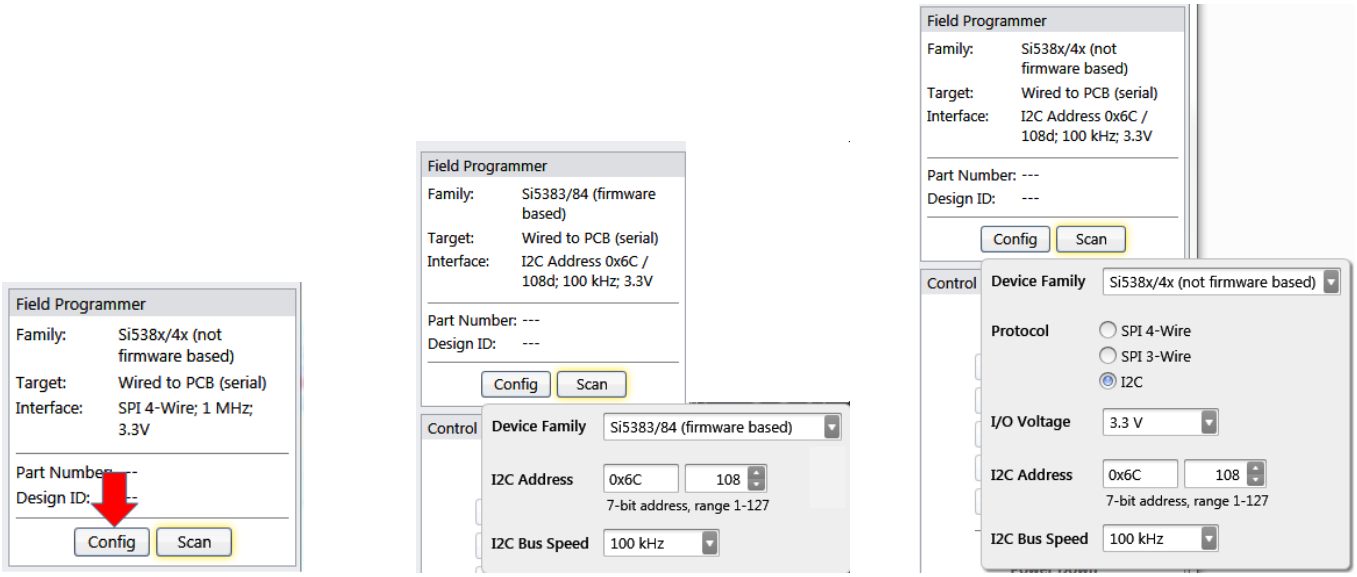

Figure 4.35. Configuring an In-system Device

After the configuration is complete, click the Scan button. The Part Number and Design ID fields should update with the device information along with the Info tab fields, as shown in Figure 4.36 In-System Scan Prompt and DUT Register Editor Tab on page 28. Now the DUT Register Editor tab can be used to make volatile register value changes to the device and the Status Registers tab can be used to monitor the status of the device.

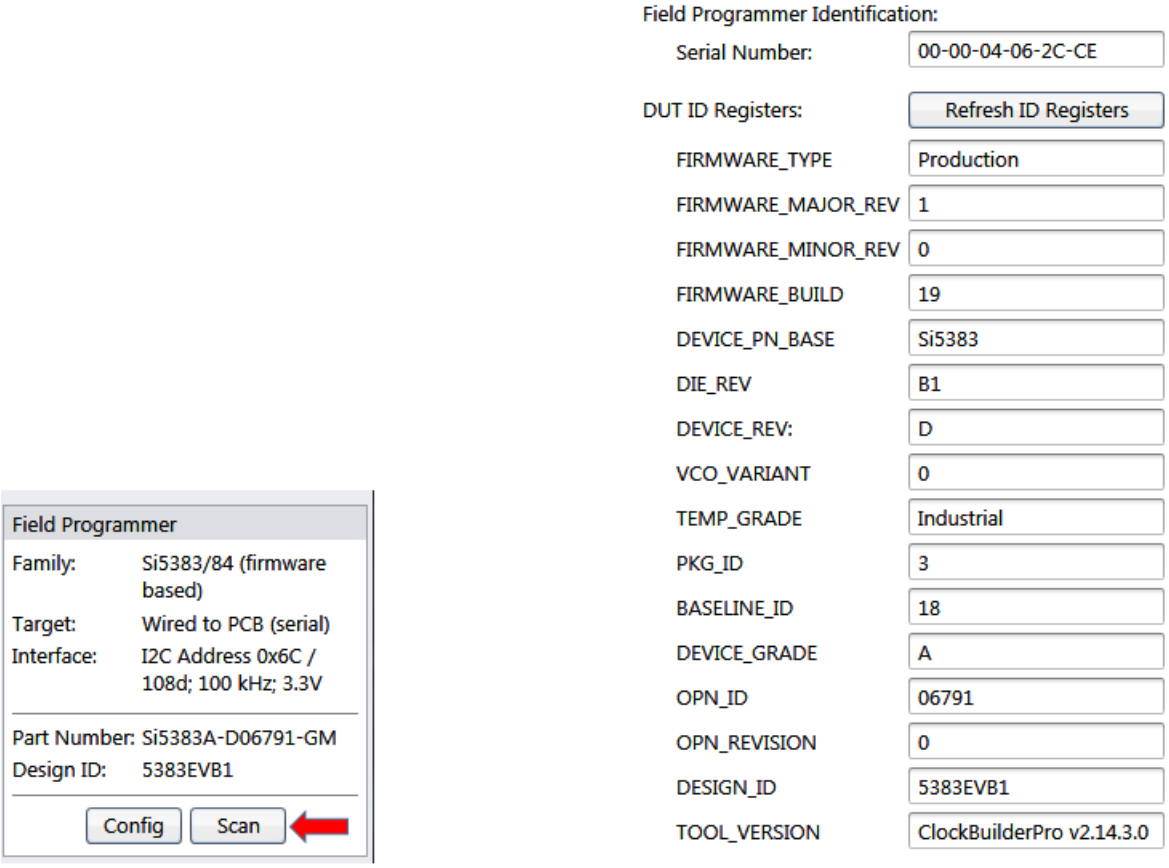

Figure 4.36. In-System Scan Prompt and DUT Register Editor Tab

4.3.5 Using the EVB GUI with In-socket Devices

CBPro will detect the connected socket when the EVB GUI is started. Click the Config button to configure the communication protocol, address (I2C), bus speed, and the I/O voltage (non-firmware based devices), as shown in the figure below.

Note: For firmware based devices the communication protocol available is I2C with a 3.3 volt I/O voltage. For non-firmware based devices, there is a selection of SPI 4-wire, SPI 3-wire, or I2C and the I/O voltage must be selected.

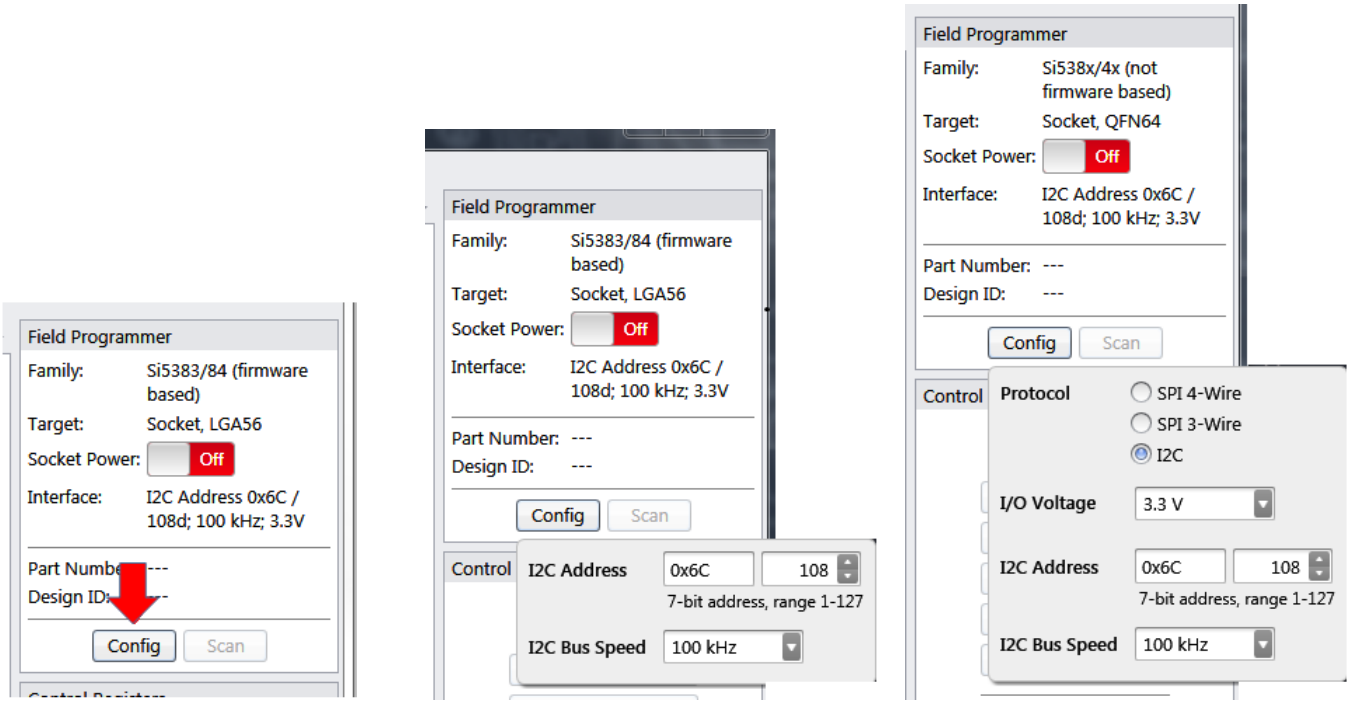

Figure 4.37. Configuring an In-socket Device

After the configuration is complete, click the Socket Power slider and the Scan button. The Part Number and Design ID fields should update with the device information along with the Info tab fields, as shown in Figure 4.38 In-Socket Scan Prompt and DUT Register Editor Tab on page 30. Now the DUT Register Editor tab can be used to make volatile register value changes to the device and the Status Registers tab can be used to monitor the status of the device.

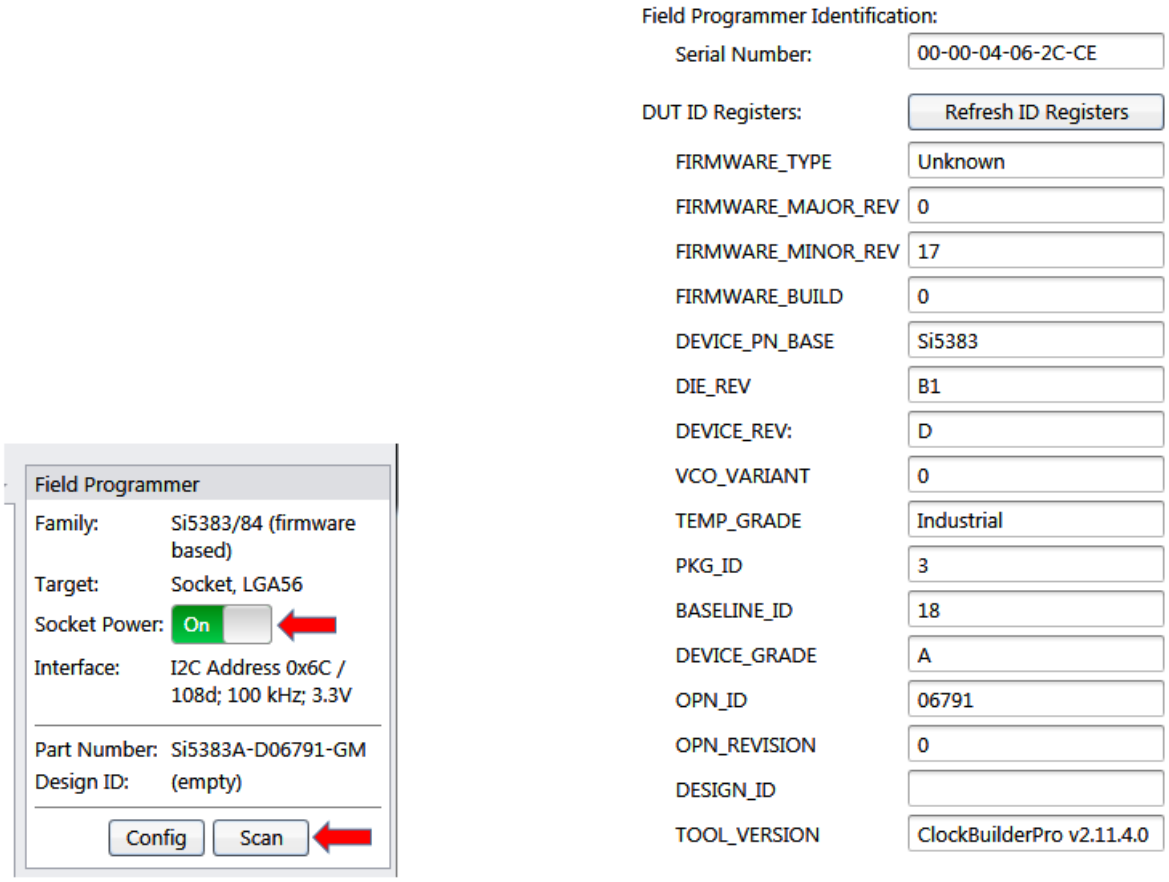

Figure 4.38. In-Socket Scan Prompt and DUT Register Editor Tab

5. CBPROG-DONGLE Schematic

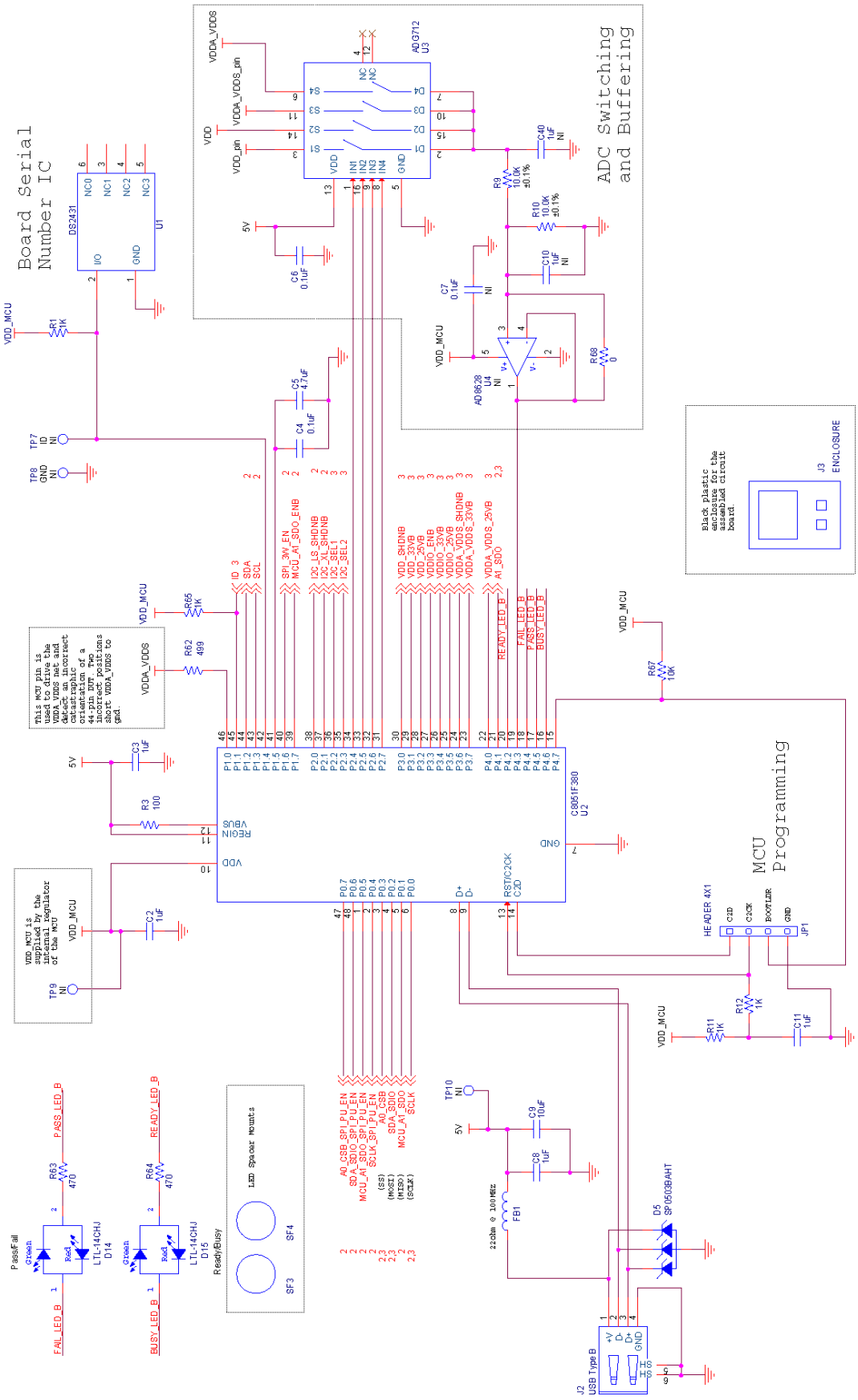

Figure 5.1. CBPROG-DONGLE Schematic (1 of 3)

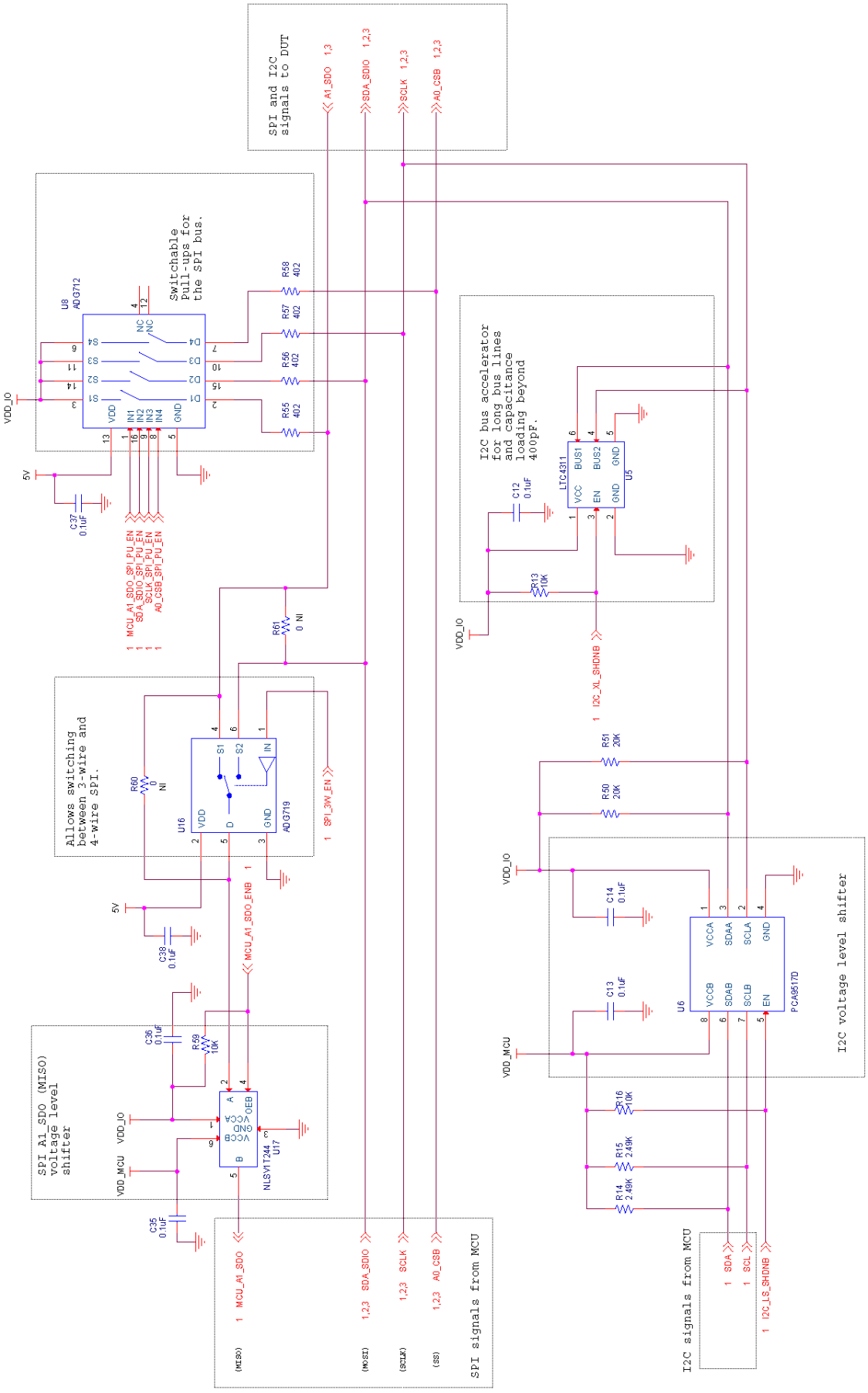

Figure 5.2. CBPROG-DONGLE Schematic (2 of 3)

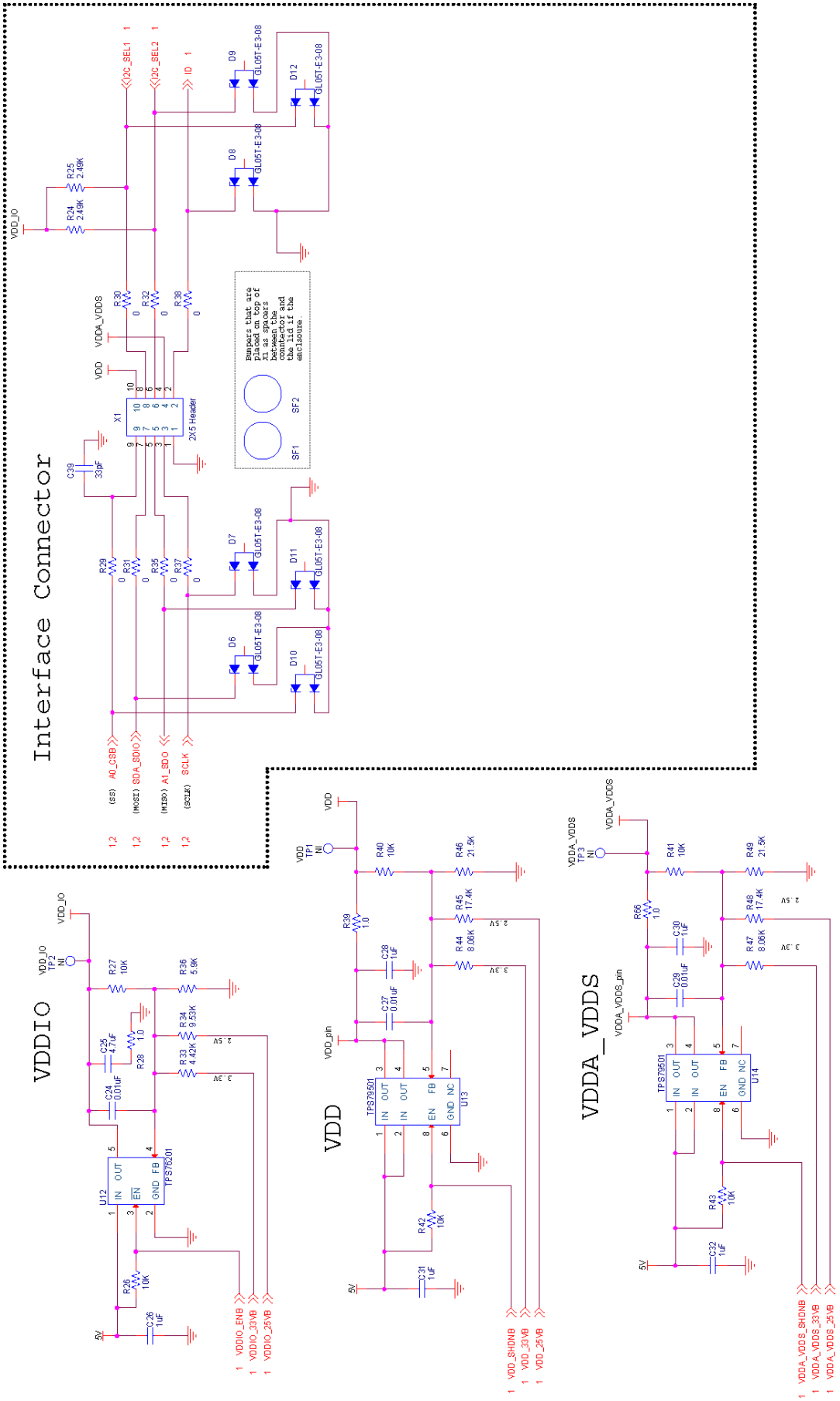

Figure 5.3. CBPROG-DONGLE Schematic (3 of 3)

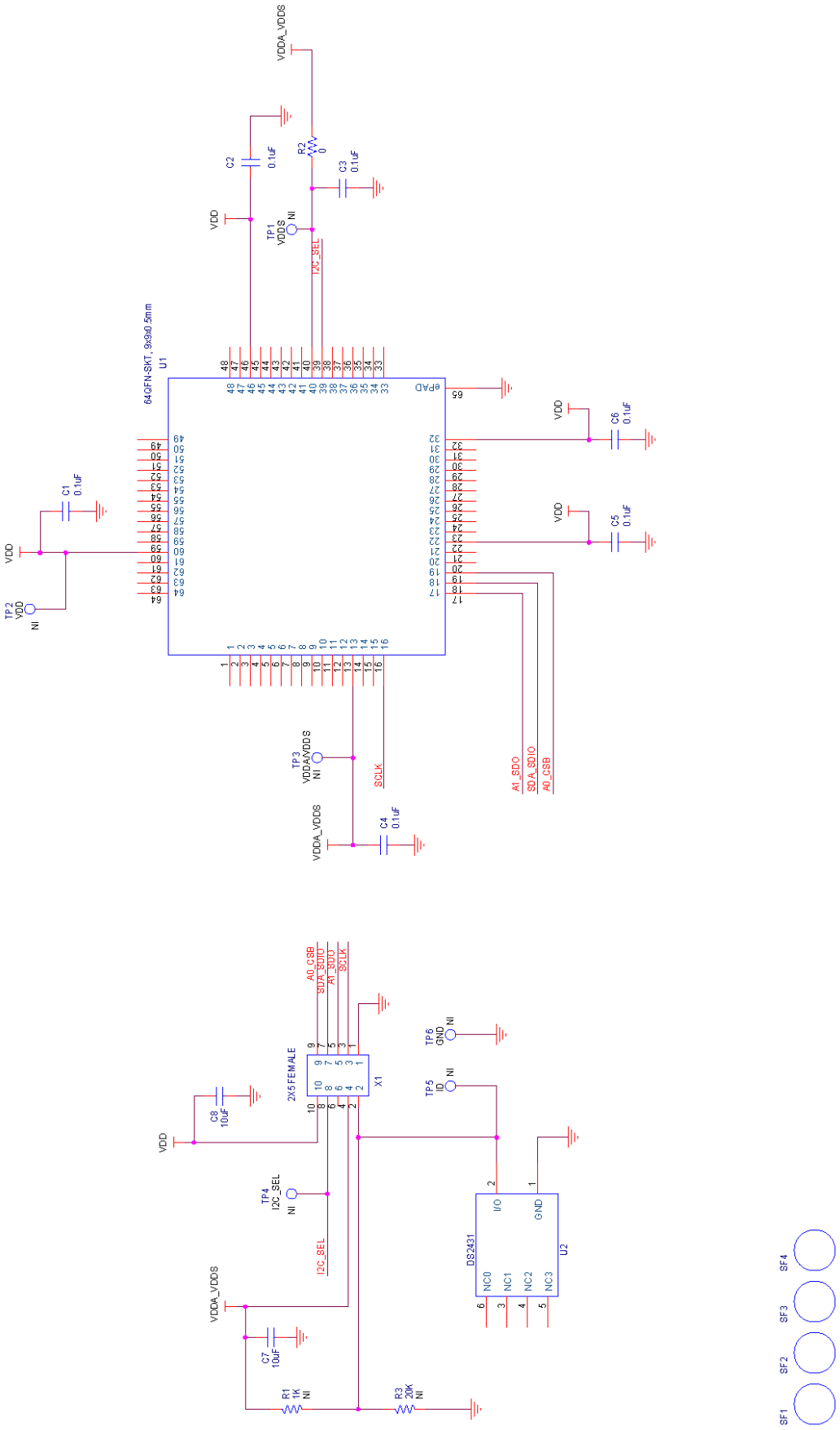

Figure 5.4. 64-Pin Socket Board Schematic

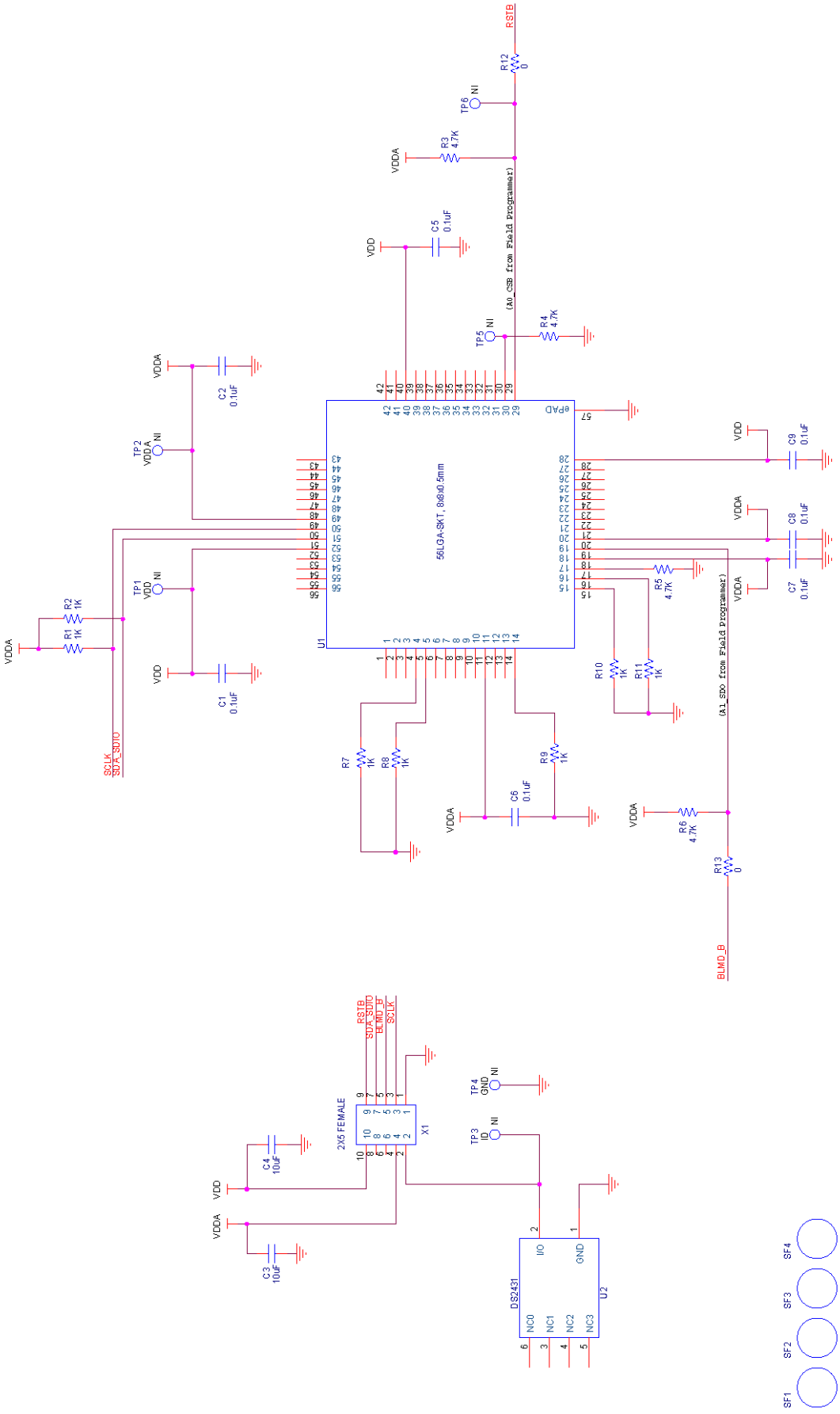

Figure 5.5. 56-Pin Socket Board Schematic

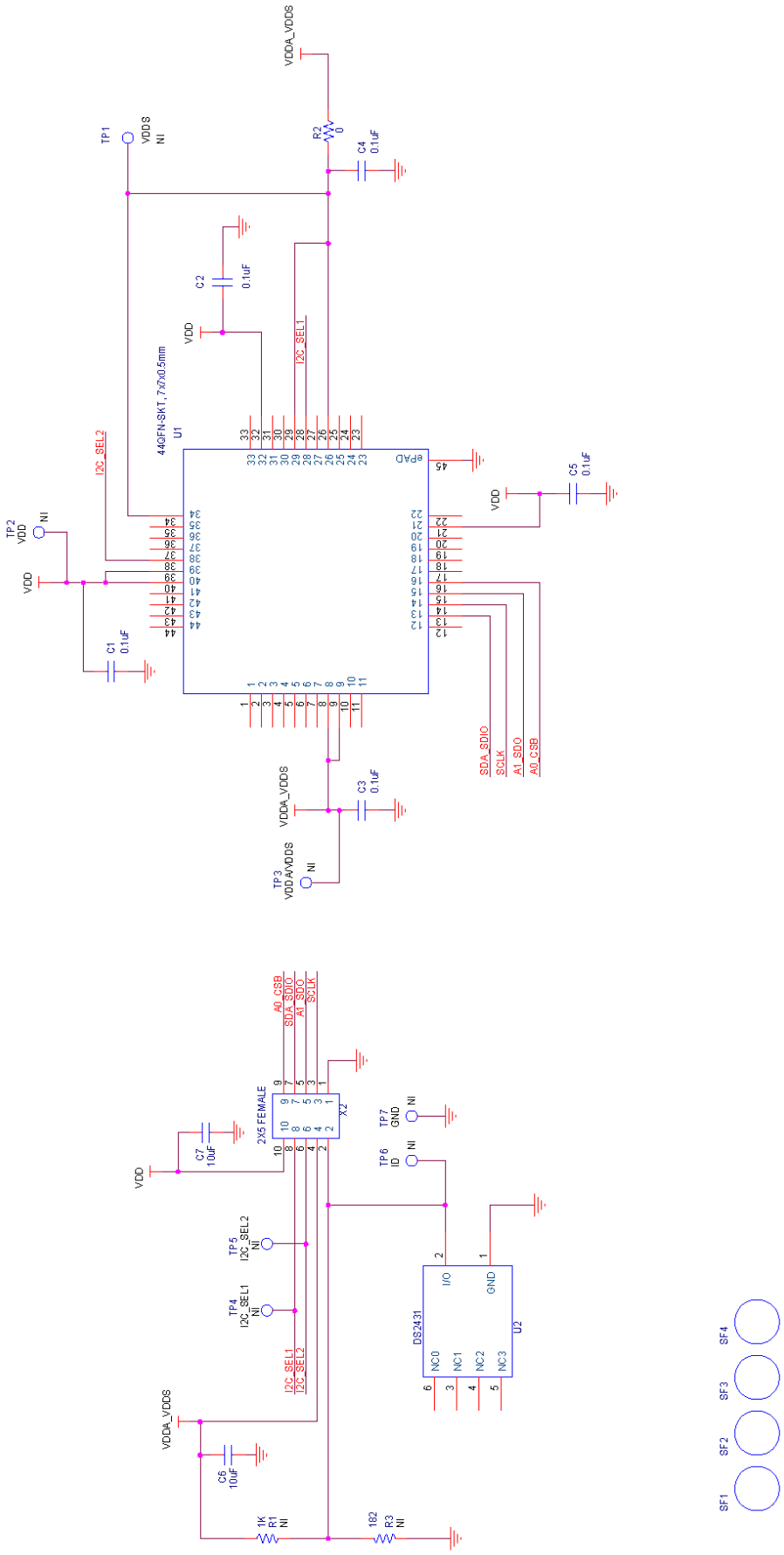

Figure 5.6. 44-Pin Socket Board Schematic

6. Bill of Materials

6.1 CBPROG-DONGLE Bill of Materials

NI	Quantity	Reference	Value	Rating	Voltage	Tolerance	Type	PCB_Footprint	ManufacturerPN	Manufacturer
	9	C2 C3 C8 C11 C26 C28 C30 C31 C32	1uF		16V	±10%	X7R	C0603	C0603X7R160-105K	Venkel
	3	C24 C27 C29	0.01uF		16V	±20%	X7R	C0603	C0603X7R160-103M	Venkel
	1	C39	33pF		25V	±10%	C0G	C0402	C0402C0G250-330K	Venkel
	9	C4 C6 C12 C13 C14 C35 C36 C37 C38	0.1uF		10V	±10%	X7R	C0402|C0402L	C0402X7R100-104K	Venkel
	2	C5 C25	4.7uF		10V	±20%	X7R	C1206	C1206X7R100-475M	Venkel
	1	C9	10uF		10V	±20%	X7R	C1206	C1206X7R100-106M	Venkel
	2	D14 D15	LTL-14CHJ	20mA				LED-T1-KK	LTL-14CHJ	LITE-ON TECHNOLOGY CORP
	1	D5	SP0503BAHT	300mW	20V		TVS	SOT143-AKKK|SOT143	SP0503BAHTG	Littlefuse
	7	D6 D7 D8 D9 D10 D11 D12	GL05T-E3-08	5A	11V		Dual Common Anode	SOT23-123	GL05T-E3-08	Vishay
	1	FB1	22 Ohm	6000mA			SMT	L0805	BLM21PG220SN1	MuRata
	1	J2	USB Type B				USB	CONN-USB-B	61729-0010BLF	FCI
	1	J3	ENCLOSURE					N/A	Emulator7045	Shanghai Zhongxingda Electronics
	4	R1 R11 R12 R65	1K	1/16W		±1%	ThickFilm	R0402	CR0402-16W-1001F	Venkel
	10	R13 R16 R26 R27 R40 R41 R42 R43 R59 R67	10K	1/16W		±1%	ThickFilm	R0402|R0402L	CR0402-16W-1002F	Venkel
	4	R14 R15 R24 R25	2.49K	1/16W		±1%	ThickFilm	R0402	CR0402-16W-2491F	Venkel
	1	R28	1.0	1/16W		±1%	ThickFilm	R0402	CR0402-16W-1R00F	Venkel
	8	R29 R30 R31 R32 R35 R37 R38 R68	0	1A			ThickFilm	R0402|R0402L	CR0402-16W-000	Venkel
	1	R3	100	1/16W		±1%	ThickFilm	R0402	CR0402-16W-1000F	Venkel
	1	R33	4.42K	1/16W		±1%	ThickFilm	R0402	CR0402-16W-4421F	Venkel
	1	R34	9.53K	1/16W		±1%	ThickFilm	R0402	CR0402-16W-9531F	Venkel
	1	R36	5.9K	1/16W		±1%	ThickFilm	R0402	CR0402-16W-5901F	Venkel
	2	R39 R66	1.0	3/4W		±1%	ThickFilm	R1210	CRCW12101R00FKEAHP	Vishay Dale
	2	R44 R47	8.06K	1/16W		±0.1%	±25PPM	R0402	TFCR0402-16W-E-8061B	Venkel
	2	R45 R48	17.4K	1/16W		±1%	ThickFilm	R0402	CR0402-16W-1742F	Venkel
	2	R46 R49	21.5K	1/16W		±1%	ThickFilm	R0402	CR0402-16W-2152F	Venkel
	2	R50 R51	20K	1/10W		±1%	ThickFilm	R0603	CR0603-10W-2002F	Venkel
	4	R55 R56 R57 R58	402	1/16W		±1%	ThickFilm	R0402	CR0402-16W-4020F	Venkel
	1	R62	499	1/16W		±1%	ThickFilm	R0402|R0402L	CR0402-16W-4990F	Venkel
	2	R63 R64	470	1/16W		±5%	ThickFilm	R0402	CR0402-16W-471J	Venkel
	2	R9 R10	10.0K	1/10W		±0.1%	±25PPM	R0603	ERA-3AEB103V	Panasonic
	2	SF1 SF2	BUMPER					RUBBER_FOOT_0.250"	SJ5382	3M
	2	SF3 SF4	SPACER					N/A	7363	Keystone Electronics
	1	U1	DS2431					SOJ6N4.45P1.27	DS2431P+	Maxim
	1	U12	TPS76201	100mA			LDO	SOT5N2.8P0.95	TPS76201DBV	TI
	2	U13 U14	TPS79501	500mA			LDO	DFN8N3.0P0.65E2.4X1.65	TPS79501DRBT	TI
	1	U16	ADG719					SOT6N2.8P0.95	ADG719BRTZ	Analog Devices
	1	U17	NLSV1T244		.9-4.5V		Buffer	UDFN6N1P0.4	NLSV1T244MUTBG	On Semi
	1	U2	C8051F380				MCU	QFP48N9X9P0.5	CF380P1104AGQ	SiLabs
	2	U3 U8	ADG712					TSSOP16N6.4P0.65	ADG712BRU	Analog Devices
	1	U5	LTC4311		5.5V			SC70-6N2.1P0.65	LTC4311CSC6#TRMPBF	Linear Technology
	1	U6	PCA9517D				I2C	SO8N6.0P1.27	PCA9517D	NXP
	1	X1	2X5 Header				Shrouded	CONN2X5-RA-SBH11	SBH11-PBPC-D05-RA-BK	Sullins Connector Solutions

Not Installed Components

NI	Quantity	Reference	Value	Rating	Voltage	Tolerance	Type	PCB_Footprint	ManufacturerPN	Manufacturer
NI	2	C10 C40	1uF		16V	±10%	X7R	C0603	C0603X7R160-105K	Venkel
NI	1	C7	0.1uF		10V	±10%	X7R	C0402|C0402L	C0402X7R100-104K	Venkel
NI	1	JP1	HEADER 4X1				Header	CONN-1X4	TSW-104-07-T-S	Samtec
NI	2	R60 R61	0	1A			ThickFilm	R0603	CR0603-16W-000	Venkel
NI	5	TP1 TP2 TP3 TP9 TP10	RED				Loop	TESTPOINT	151-207-RC	Kobiconn
NI	1	TP7	BLUE				Loop	TESTPOINT	151-205-RC	Kobiconn
NI	1	TP8	BLACK				Loop	TESTPOINT	151-203-RC	Kobiconn
NI	1	U4	AD8628		5V		OPAMP	SOT23-5N	AD8628AUJ-R2	Analog Devices

6.2 Si538x4x-64SKT-DK Socket Board BOM

NI	Quantity	Reference	Value	Rating	Voltage	Tolerance	Type	PCB_Footprint	ManufacturerPN	Manufacturer
	6	C1 C2 C3 C4 C5 C6	0.1uF		10V	±10%	X7R	C0402|C0402L	C0402X7R100-104K	Venkel
	2	C7 C8	10uF		10V	±20%	X7R	C1206	C1206X7R100-106M	Venkel
	1	R2	0	1A			ThickFilm	R0402|R0402L	CR0402-16W-000	Venkel
	4	SF1 SF2 SF3 SF4	BUMPER					RUBBER_FOOT_SMALL	SJ61A6	3M
	1	U1	64QFN-SKT, 9x9x0.5mm				QFN	QFN64N9X9P0.5-SKT-WELLS-CTI	790-42064-101G	Sensata
	1	U2	DS2431					SOJ6N4.45P1.27	DS2431P+	Maxim
	1	X1	2X5 FEMALE				CONN	CONN2X5-FRA-SFH11	SFH11-PBPC-D05-RA-BK	Sullins Connector Solutions

Not Installed Components

NI	Quantity	Reference	Value	Rating	Voltage	Tolerance	Type	PCB_Footprint	ManufacturerPN	Manufacturer
NI	1	R1	1K	1/16W		±1%	ThickFilm	R0402	CR0402-16W-1001F	Venkel
NI	1	R3	20K	1/16W		±1%	ThickFilm	R0402	CR0402-16W-2002F	Venkel
NI	3	TP1 TP2 TP3	RED				Loop	TESTPOINT	151-207-RC	Kobiconn
NI	2	TP4 TP5	BLUE				Loop	TESTPOINT	151-205-RC	Kobiconn
NI	1	TP6	BLACK				Loop	TESTPOINT	151-203-RC	Kobiconn

6.3 Si538x4x-56SKT-DK Socket Board Bill of Materials

NI	Quantity	Reference	Value	Rating	Voltage	Tolerance	Type	PCB_Footprint	ManufacturerPN	Manufacturer
	7	C1 C2 C5 C6 C7 C8 C9	0.1uF		10V	±10%	X7R	C0402|C0402L	C0402X7R100-104K	Venkel
	2	C3 C4	10uF		10V	±20%	X7R	C1206	C1206X7R100-106M	Venkel
	7	R1 R2 R7 R8 R9 R10 R11	1K	1/16W		±1%	ThickFilm	R0603	CR0603-16W-1001F	Venkel
	2	R12 R13	0	1A			ThickFilm	R0603|R0603L	CR0603-16W-000	Venkel
	4	R3 R4 R5 R6	4.7K	1/10W		±1%	ThickFilm	R0603	CR0603-10W-4701F	Venkel
	4	SF1 SF2 SF3 SF4	BUMPER					RUBBER_FOOT_SMALL	SJ61A6	3M
	1	U1	56LGA-SKT, 8x8x0.5mm				LGA	QFN56N8X8P0.5-SKT-WELLS-CTI	790-42056-101G	Sensata
	1	U2	DS2431					SOJ6N4.45P1.27	DS2431P+	Maxim
	1	X1	2X5 FEMALE				CONN	CONN2X5-FRA-SFH11	SFH11-PBPC-D05-RA-BK	Sullins Connector Solutions

Not Installed Components

NI	Quantity	Reference	Value	Rating	Voltage	Tolerance	Type	PCB_Footprint	ManufacturerPN	Manufacturer
NI	2	TP1 TP2	RED				Loop	TESTPOINT	151-207-RC	Kobiconn
NI	3	TP3 TP5 TP6	BLUE				Loop	TESTPOINT	151-205-RC	Kobiconn
NI	1	TP4	BLACK				Loop	TESTPOINT	151-203-RC	Kobiconn

6.4 Si538x4x-44SKT-DK Socket Board Bill of Materials

NI	Quantity	Reference	Value	Rating	Voltage	Tolerance	Type	PCB_Footprint	ManufacturerPN	Manufacturer
	5	C1 C2 C3 C4 C5	0.1uF		10V	±10%	X7R	C0402|C0402L	C0402X7R100-104K	Venkel
	2	C6 C7	10uF		10V	±20%	X7R	C1206	C1206X7R100-106M	Venkel
	1	R2	0	1A			ThickFilm	R0402|R0402L	CR0402-16W-000	Venkel
	4	SF1 SF2 SF3 SF4	BUMPER					RUBBER_FOOT_SMALL	SJ61A6	3M
	1	U1	44QFN-SKT, 7x7x0.5mm				QFN	QFN44N7X7P0.5-SKT-WELLS-CTI	790-41044-101G	Sensata
	1	U2	DS2431					SOJ6N4.45P1.27	DS2431P+	Maxim
	1	X2	2X5 FEMALE				CONN	CONN2X5-FRA-SFH11	SFH11-PBPC-D05-RA-BK	Sullins Connector Solutions

Not Installed Components

NI	Quantity	Reference	Value	Rating	Voltage	Tolerance	Type	PCB_Footprint	ManufacturerPN	Manufacturer
NI	1	R1	1K	1/16W		±1%	ThickFilm	R0402	CR0402-16W-1001F	Venkel
NI	1	R3	182	1/16W		±1%	ThickFilm	R0402	CR0402-16W-1820F	Venkel
NI	3	TP1 TP2 TP3	RED				Loop	TESTPOINT	151-207-RC	Kobiconn
NI	3	TP4 TP5 TP6	BLUE				Loop	TESTPOINT	151-205-RC	Kobiconn
NI	1	TP7	BLACK				Loop	TESTPOINT	151-203-RC	Kobiconn

7. Appendix A. Troubleshooting

7.1 Why can't I communicate with the Si534x8x device on my hardware using the CBPro Dongle?

There are multiple windows in the CBPro software that use or provide communication to the device connected to the CBPro Dongle. The examples below show the windows and type of errors you may encounter. All of these situations can be resolved using the following steps.

General Steps to Resolve a Communication Issue (Non-Firmware based devices)

1. Verify which communication protocol your hardware is using – SPI or I2C.
2. Verify the voltage level on the I2C_SEL control pin on the DUT. This level should be logic low (0 V) if your communication protocol is SPI. This level should be logic high (1.8 V or 3.3 V – refer step 3 below) if your communication protocol is I2C.
3. Verify the value of the IO_VDD_SEL bit (Register 0x0943[0]) for the DUT. If IO_VDD_SEL is 0, the I/O Voltage setting should be 1.8V. If IO_VDD_SEL is 1, the I/O Voltage setting should 3.3V. If you do not know this value, you can try both voltages to determine which voltage level will work successfully.
4. If the communication protocol is I2C, verify the I2C address setting (Register 0x000B) for the device. You may also need to verify the voltage level on the A0/CSb and A1/SDO pins if they are not connected to the field programmer. The level on these pins set bit 1 and bit 0 in the I2C address. If these are connected to the CBPro Dongle, they are both driven low.

General Steps to Resolve a Communication Issue (Firmware based devices)

1. Verify the I2C address for the device.
2. Verify the voltage level on the A0/CSb and A1/SDO pins if they are not connected to the field programmer. The level on these pins set bit1 and bit 0 in the I2C address. If these are connected to the CBPro Dongle, they are both driven low.

Communication Error Using the Design Dashboard Window

If the design dashboard experiences an error communicating the device, the following error window will appear.

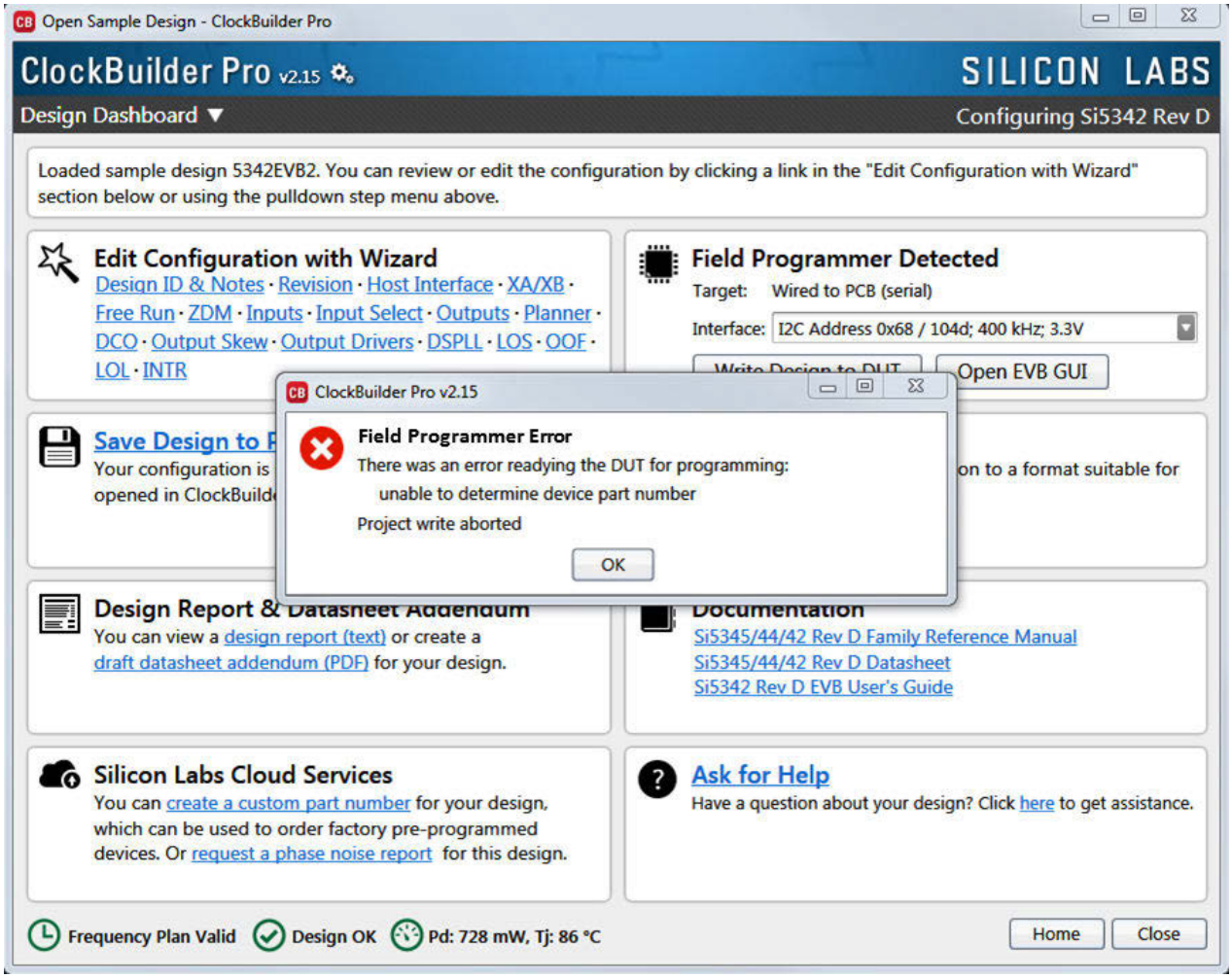

Figure 7.1. Communication Error Using Design Dashboard

This example window shows how to adjust the communication settings of the dashboard to resolve communication error.

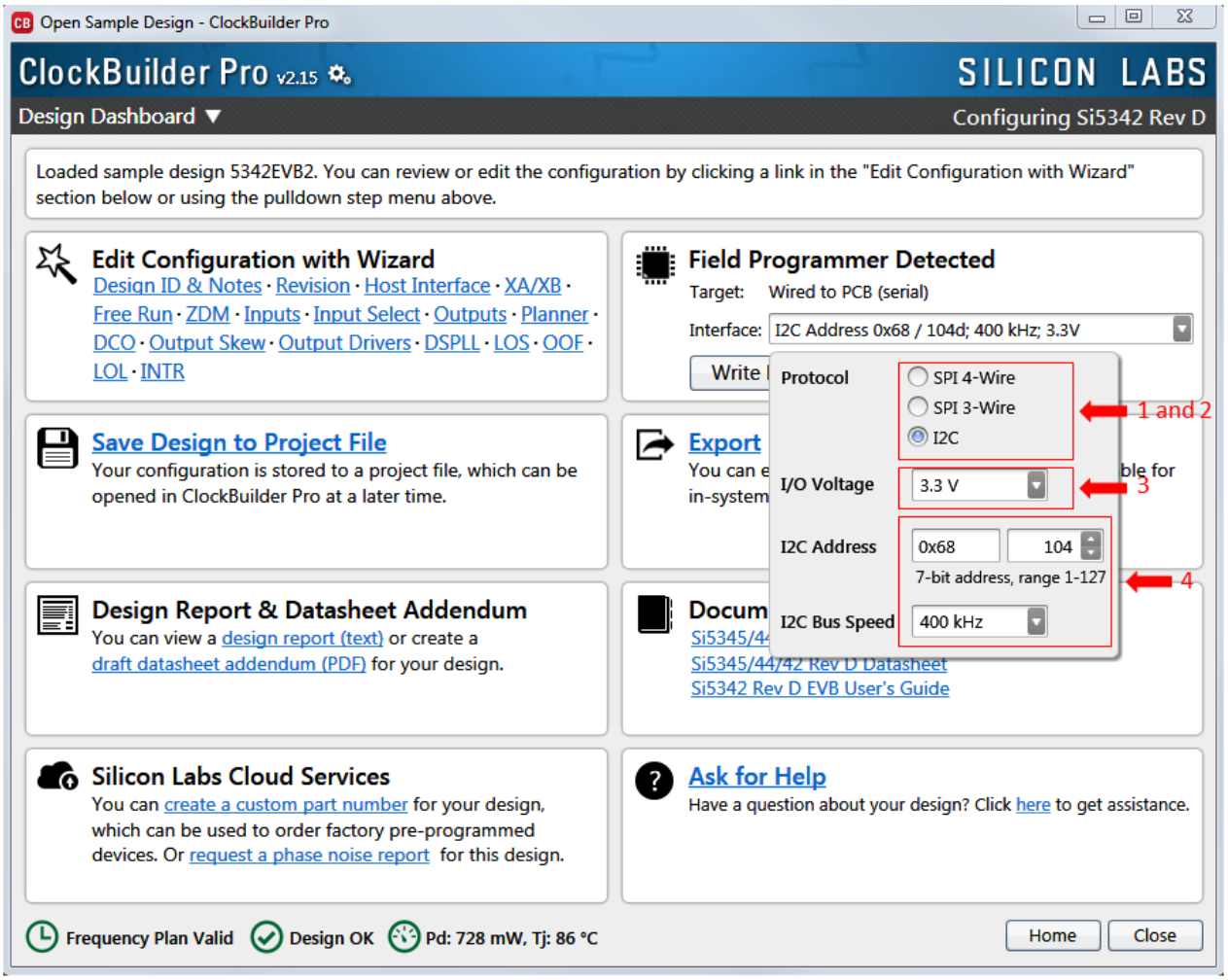

Figure 7.2. Design Dashboard Communication Error Solution

Communication Error Using the Burn NVM Window

The following window shows a communication error in the NVM Burn window. This error will appear after the Scan for Device button is pressed.

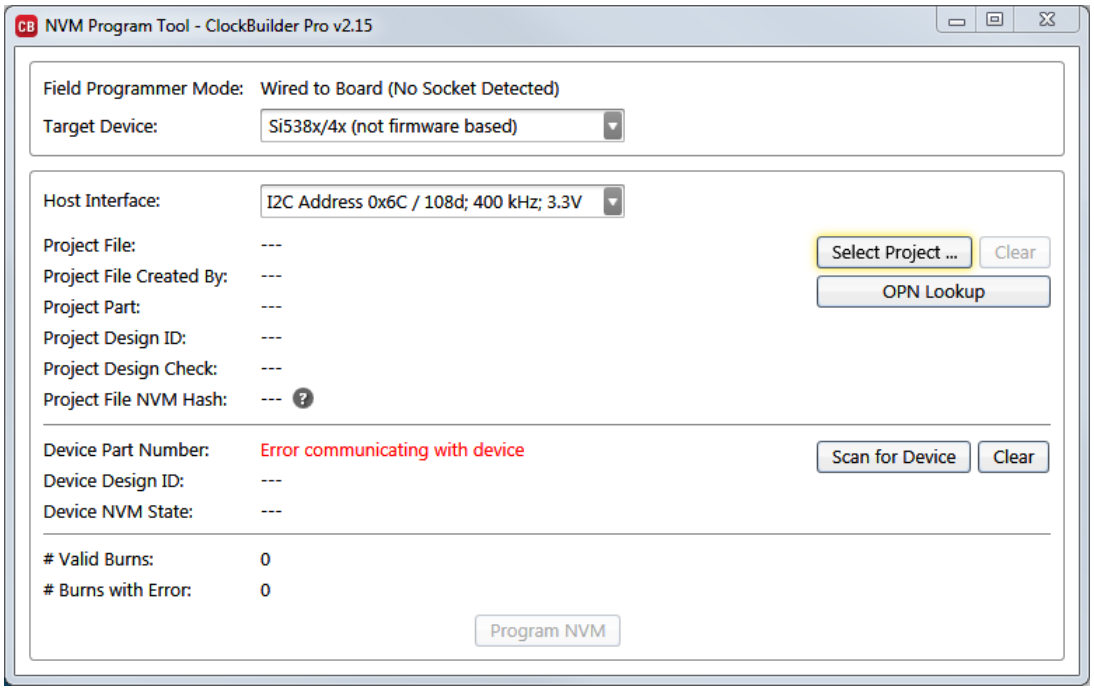

Figure 7.3. Burn NVM Error Message

The following window shows how to adjust the communication settings of the dashboard to resolve communication error.

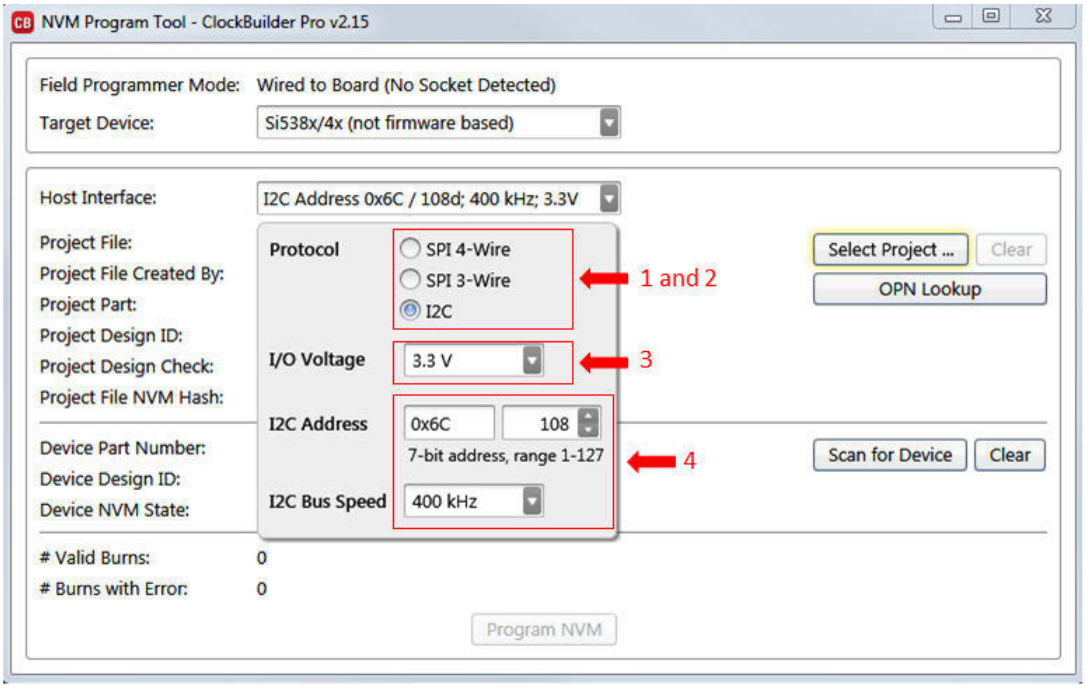

Figure 7.4. Burn NVM Error Message Solution

Communication error using the EVB GUI window

The following window shows an example of the error produced when the EVB GUI experiences an I2C error.

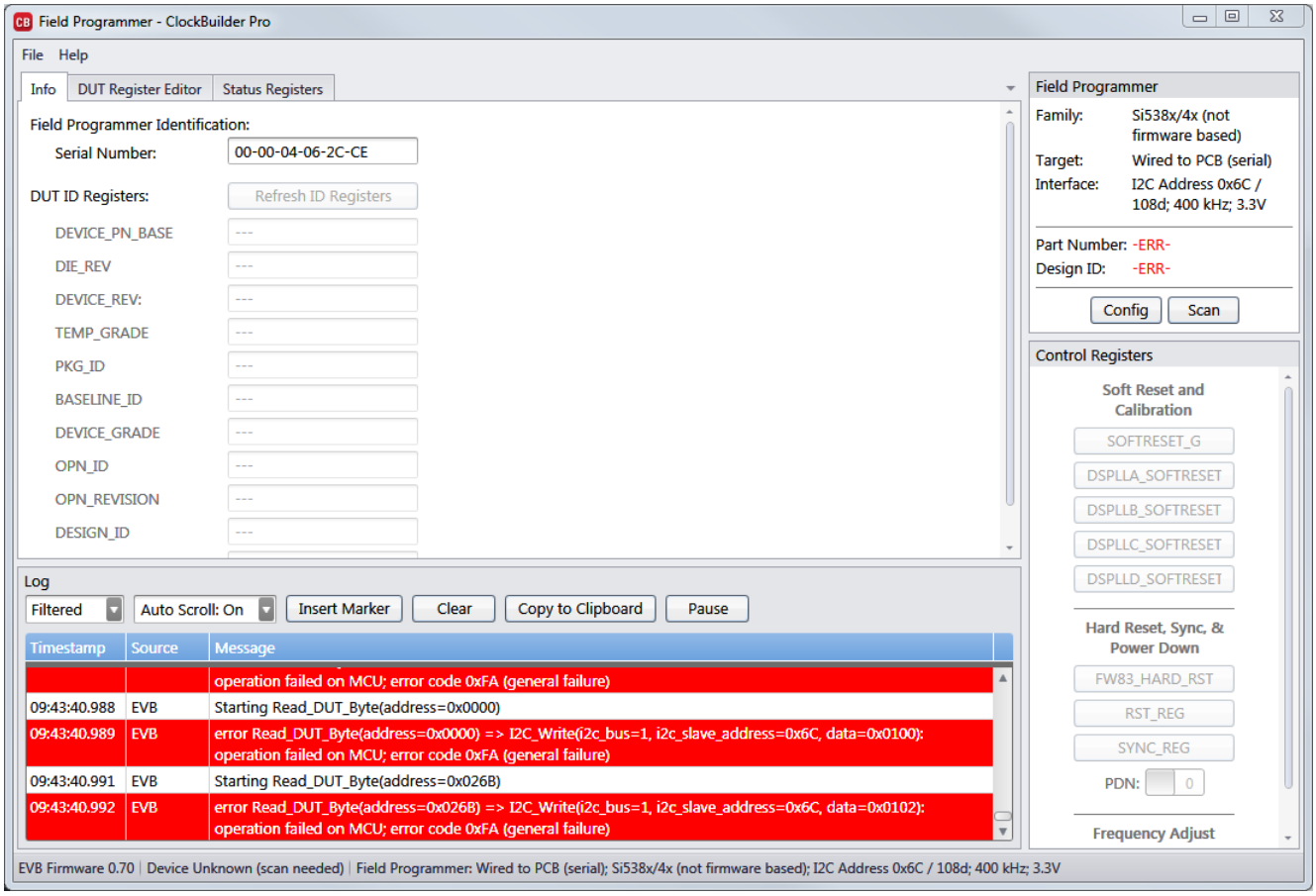

Figure 7.5. EVB GUI I2C Error

The following window shows an example of the error produced when the EVB GUI experiences an SPI error.

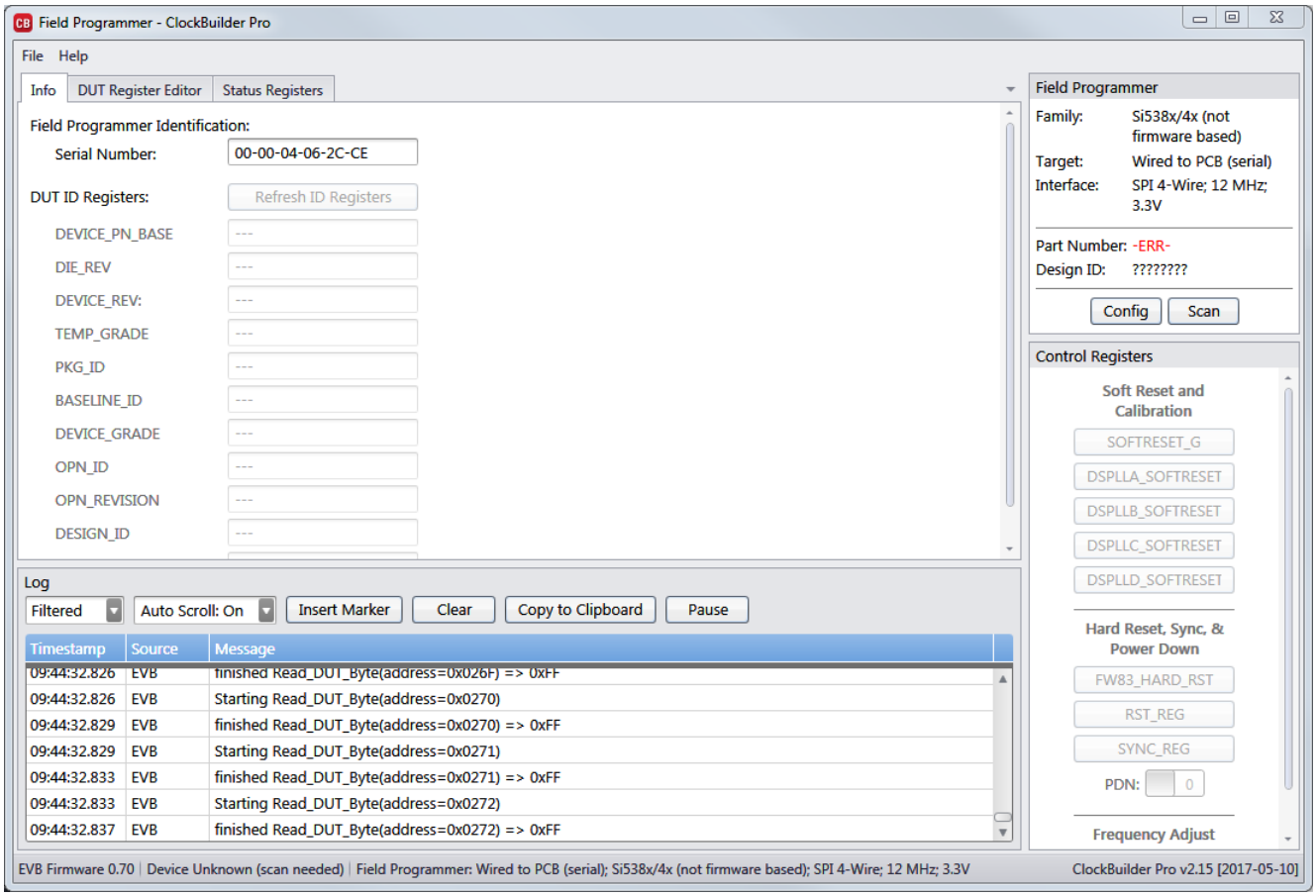

Figure 7.6. EVB GUI SPI Error

The following window shows how to change the communication settings using the EVB GUI window.

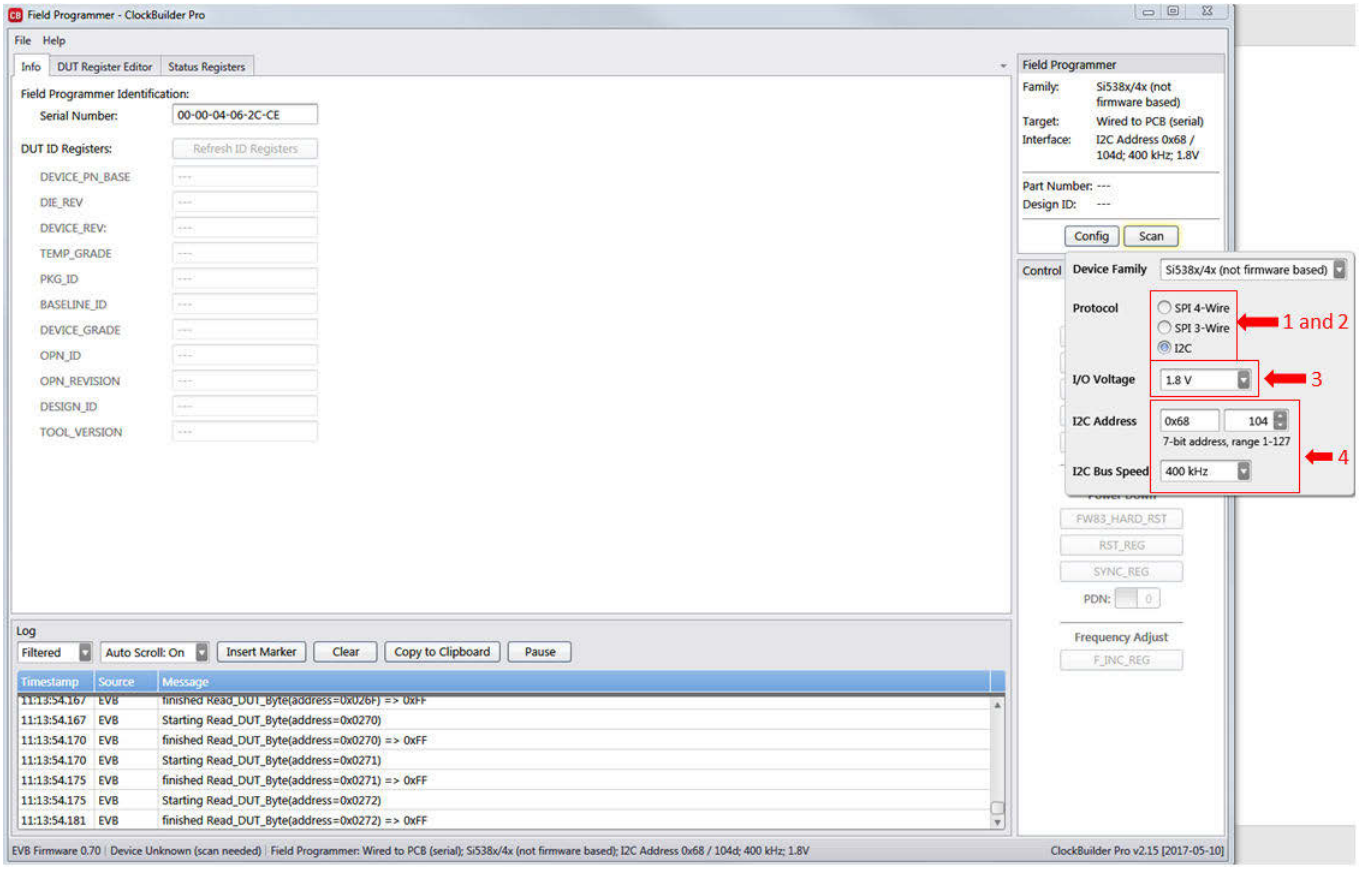

Figure 7.7. EVB GUI Solution

7.2 Why do I have a communication error when I write my new project to the device?

New Plan Changes the IO_VDD_SEL Bit (Register 0x0943[0]) Value

In order for the CBPro Dongle to communicate with the device correctly, the dongle's IO voltage needs to match the IO_VDD_SEL bit in the device. If the plan changes this bit during the writing process, communication will fail. To determine if the new plan is changing this bit, perform the following steps:

- Read the current value in the device by using the DUT Register Editor tab in the EVB GUI window.
- Determine if the new plan changes the value. This can be done by looking at the Host Interface tab in the Design Dashboard of the new project.
  - If VDD (Core) radio button selected and 0x943 = 0, no change from new plan,

    Else VDD (Core) radio button selected and 0x943 = 1, new plan is changing IO_VDD_SEL refer to 7.3 How do I write a project file to the device that changes the I/O Power Supply setting (IO_VDD_SEL bit)?
  - If VDDA (3.3 V) radio button selected and 0x943 = 1, no change from new plan,

    Else VDDA (3.3 V) radio button selected and 0x943 = 0, new plan is changing IO_VDD_SEL refer to 7.3 How do I write a project file to the device that changes the I/O Power Supply setting (IO_VDD_SEL bit)?

The following window shows how to read the IO_VDD_SEL bit from the device.

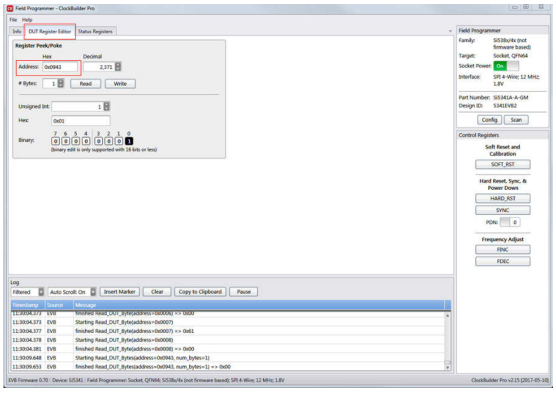

Figure 7.8. Read IO_VDD_SEL Bit from Device

The following window shows how to determine the value of the IO_VDD_SEL bit that will be written to the device from the project file.

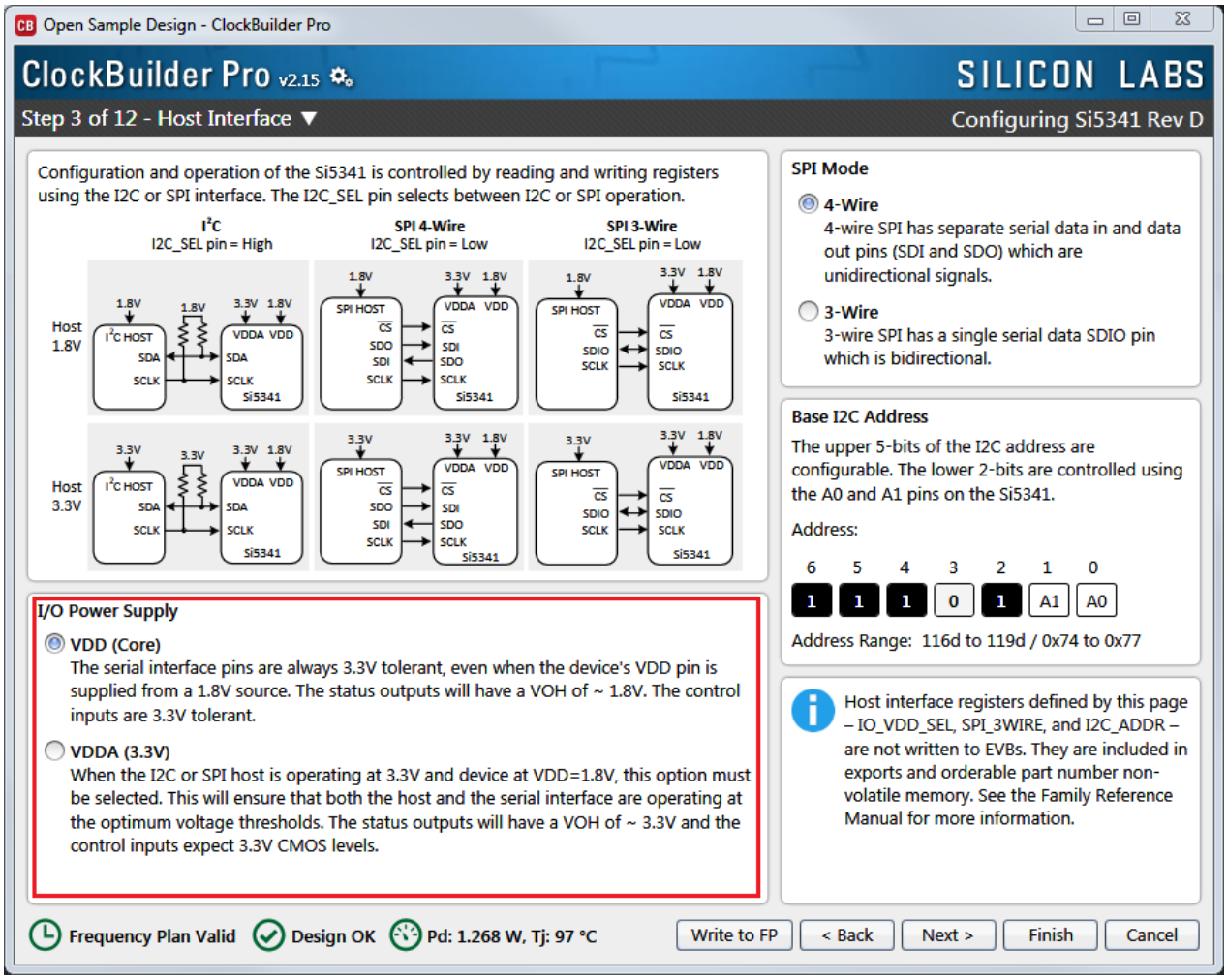

Figure 7.9. Determine the Value of IO_VDD_SEL Bit Written to Device

7.3 How do I write a project file to the device that changes the I/O Power Supply setting (IO_VDD_SEL bit)?

General Steps to Change I/O Power Supply Setting with a Project File

In order for the field programmer to communicate with the device correctly, the field programmer's IO voltage needs to match the IO_VDD_SEL bit in the device and use the correct serial communication protocol to match the I2C_SEL pin on the device. This is not automatically detected by the GUI or the CLI command.

If the new project changes the IO_VDD_SEL bit, the following summarized steps need to be performed. The flow chart and figures that follow provide the details for each of these steps. There are detailed steps using CBPro Graphical User Interface and detailed steps using the CBPro Command Line interface.

1. Establish communication with the device to be programmed and determine the current value of the IO_VDD_SEL (0x0943[0]) bit.
2. The current value of the IO_VDD_SEL bit matches the value of the new plan to be written to the device?
   - Yes – Proceed to step 3.
   - No – Change the IO_VDD_SEL bit to match the value in the new plan. Re-establish communication with the device after changing the IO_VDD_SEL value (change the field programmer I/O Voltage to match new value for IO_VDD_SEL).
3. Write the new plan to the device.

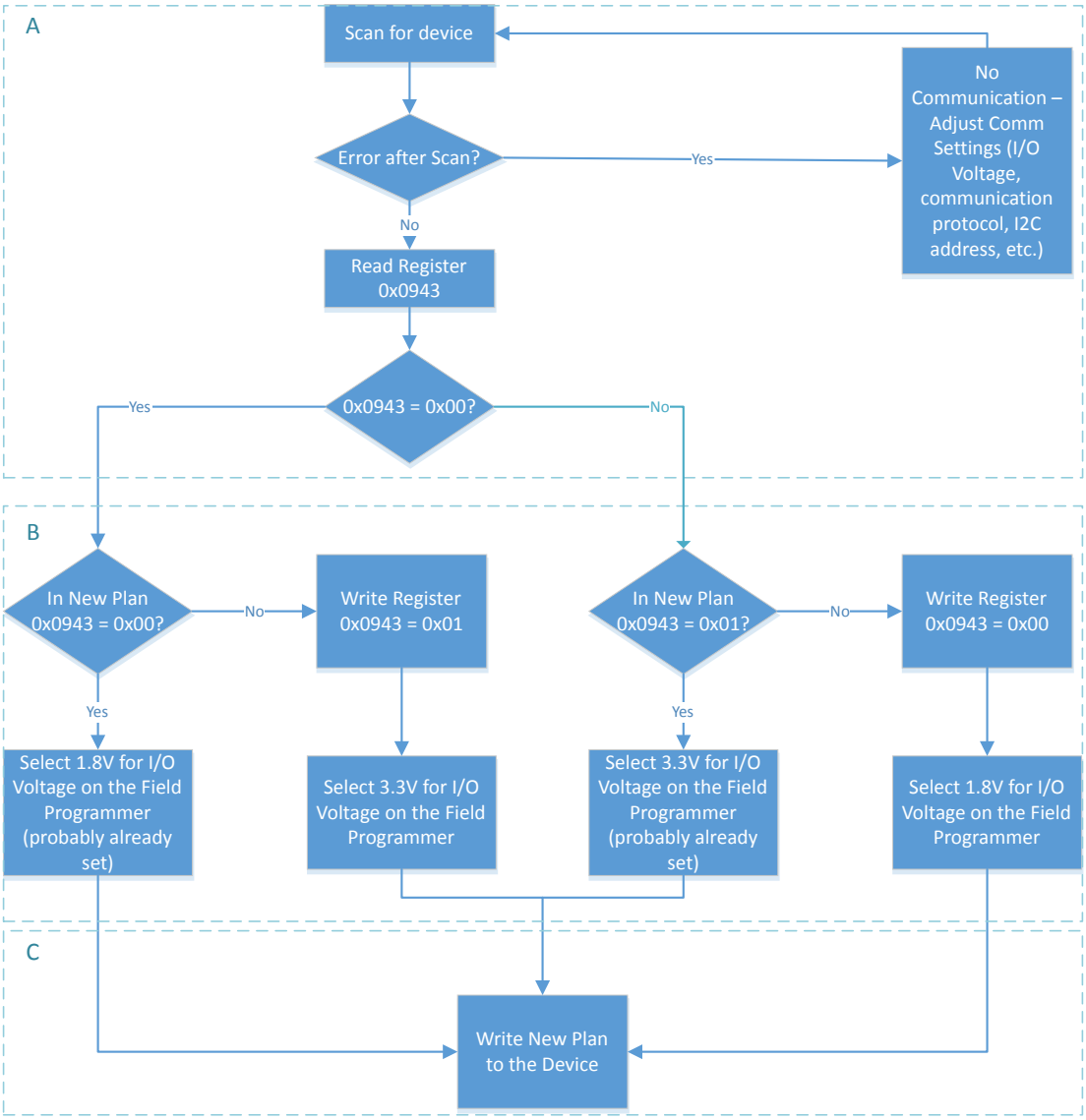

Figure 7.10. General Steps to Change I/O Power Supply Setting with a Project File

Steps using CBPro Graphical User Interface

1. Select the 'EVB GUI' button on the home screen as shown to attempt communication with the device.

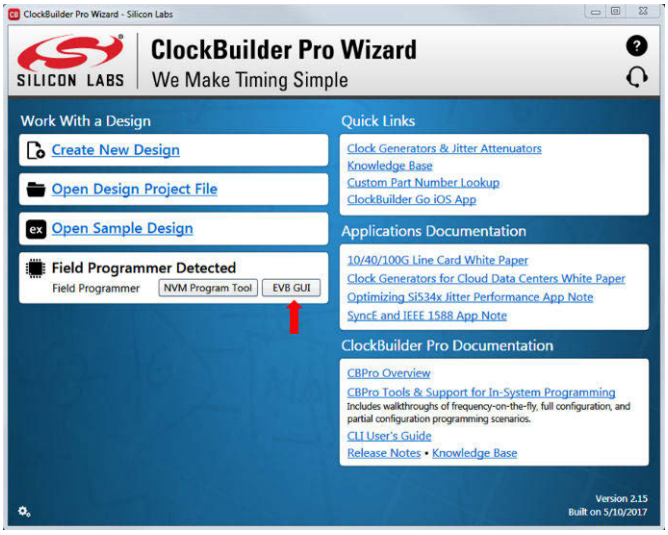

Figure 7.11. EVB GUI Button

a. Select the 'DUT Register Editor' tab.

b. Determine the correct device communication protocol and setup CBPro accordingly as shown. For an In-socket device, click the Socket Power slider to power up the device. For In-system devices, click the Device Family pulldown and select the appropriate device family.

c. Click the Scan button to verify communication with the device.

d. If communication is successful, the device part number and design ID will be updated. If communication is not successful, the part number field will display -ERR- and the DUT register tab will be disabled.

Configuring communication settings:

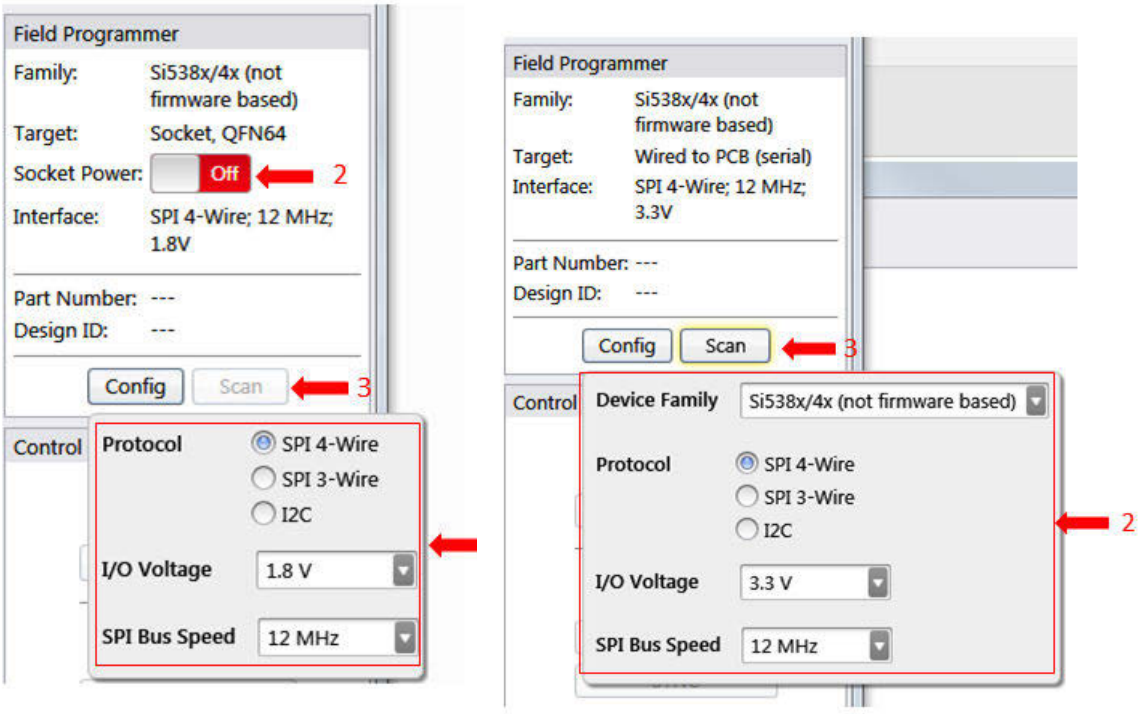

Figure 7.12. Configuring Communication Settings

Examples of a Communication failure for I2C and SPI:

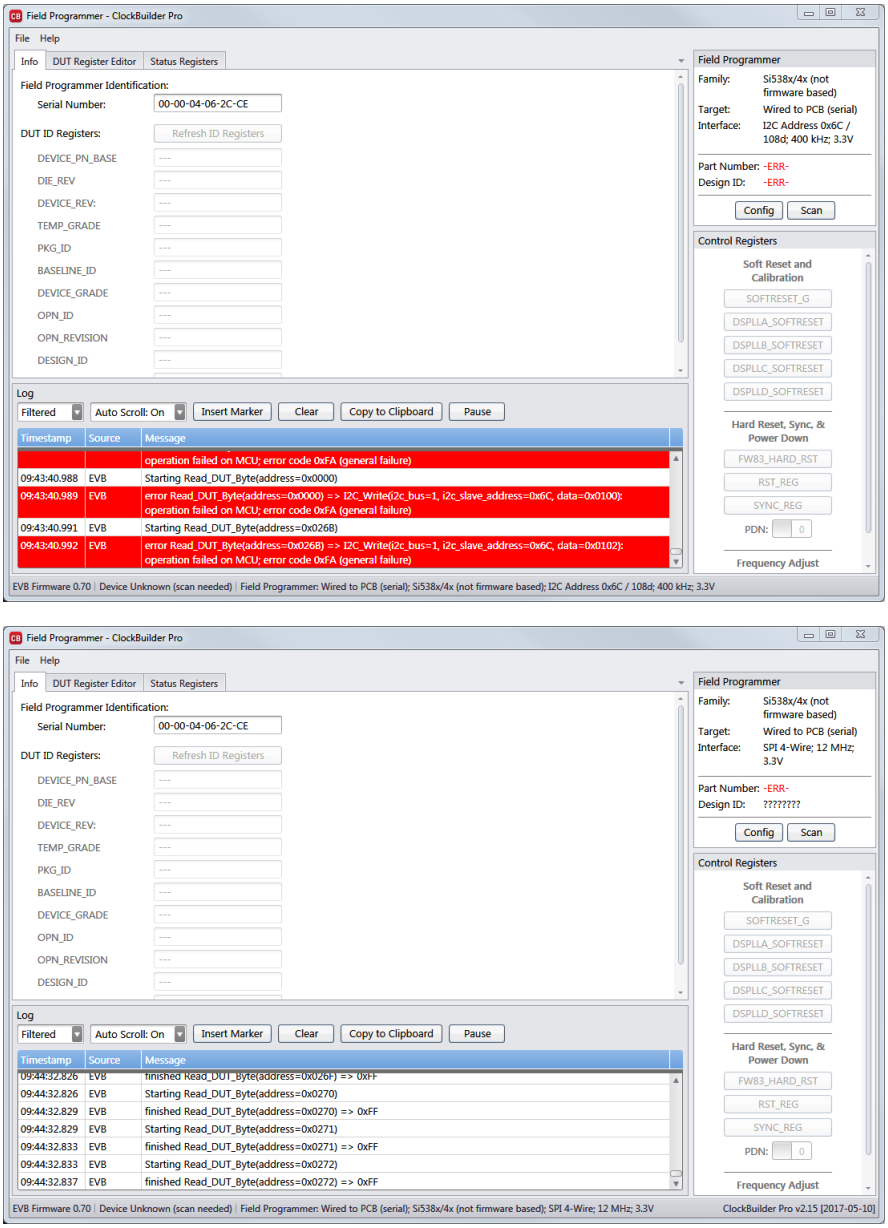

Figure 7.13. I2C and SPI Communication Failure Examples

2. Match the IO_VDD_SEL bit to the value in the plan that will be written to the device.
   a. If the IO_VDD_SEL bit already matches the value in the plan to be written, skip to step 3.
   b. If the IO_VDD_SEL bit is not correct, change the value and write the new value to the device (see the figure below).
   c. Re-configure the communication settings of the field programmer to re-establish communication to the device.

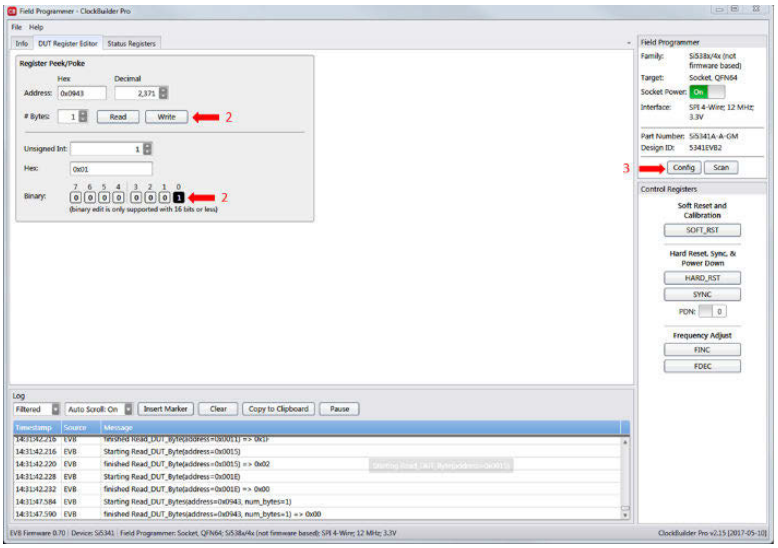

Figure 7.14. Re-configuring Communication Settings of the Field Programmer

3. Write your new plan to the device.

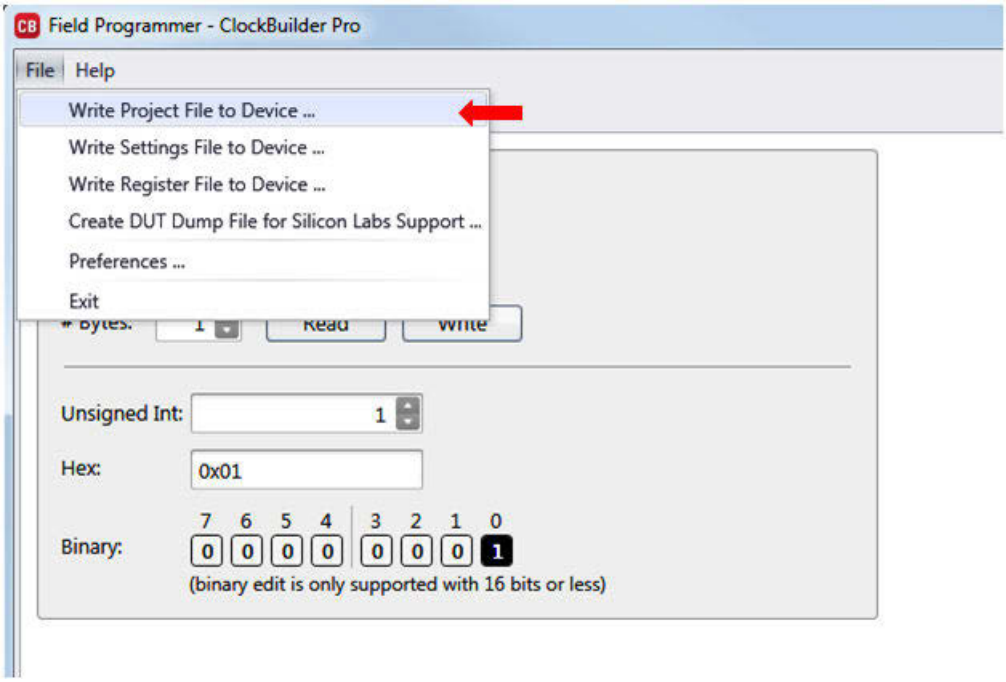

Figure 7.15. Write New Plan to Device

Steps using CBPro Command Line Interface

1. Attempt to communicate with the Si534x8x device and determine the current value of the IO_VDD_SEL bit.

   SPI communication Examples:

```
CBProDeviceRead.exe --io-voltage 1.8 --mode spi4wire --speed 1M --family si538x4x --registers 0x0943

CBProDeviceRead.exe --io-voltage 3.3 --mode spi4wire --speed 1M --family si538x4x --registers 0x0943
```

   Note: The commands above are examples. Refer to the document and help for the CBPro CLI for your specific configuration.

   I2C communication Examples:

```
CBProDeviceRead.exe --io-voltage 1.8 --mode i2c --speed 100k --i2c-address 0x68 --family si538x4x --registe
rs 0x0943

CBProDeviceRead.exe --io-voltage 3.3 --mode i2c --speed 100k --i2c-address 0x68 --family si538x4x --registe
rs 0x0943
```

   Note: The commands above are examples. Refer to the document and help for the CBPro CLI for your specific configuration.

2. Match the IO_VDD_SEL bit to the value in the plan that will be written to the device.
   a. A simple text file will need to be created that will write register 0x943 to 0x00 or 0x01.

      To write 0x01 to 0x0943, the text file should contain the following single line of text:

```
0x0943,0x01
```

      To write 0x00 to 0x0943, the text file should contain the following single line of text:

```
0x0943,0x00
```

   b. Run the CLI command below to change the IO_VDD_SEL bit.

      SPI Example:

```
CBProDeviceWrite.exe --mode spi4wire --speed 4M --io-voltage 3.3 --family si538x4x --registers simple_t
ext_file.txt
```

      I2C Example:

```
CBProDeviceWrite.exe --mode i2c –i2c-address 0x68 --speed 400K --io-voltage 3.3 --family si538x4x --reg
isters simple_text_file.txt
```

      Note: The commands above are examples. Refer to the document and help for the CBPro CLI for your specific configuration.

3. Write the new plan to the part.

   SPI Example:

```
CBProDeviceWrite.exe --mode spi4wire --speed 4M --io-voltage 3.3 --family si538x4x --project your_plan_name
.slabtimeproj
```

   I2C Example:

```
CBProDeviceWrite.exe --mode i2c –i2c-address 0x68 --speed 400K --io-voltage 3.3 --family si538x4x --projec
t your_plan_name.slabtimeproj
```

   Note: The commands above are examples. Refer to the document and help for the CBPro CLI for your specific configuration.

7.4 I burned a project file to my device with a new Base I2C address, but the base address in the device was not changed after the burn process was complete.

The I2C address will not be changed during the burn process. Changes to the base I2C address in the CBPro Configuration Wizard will be included in exports and the project file used to create orderable part numbers. However, this change is not burned to the device using the NVM Burn Tool. See the note highlighted in the figure below.

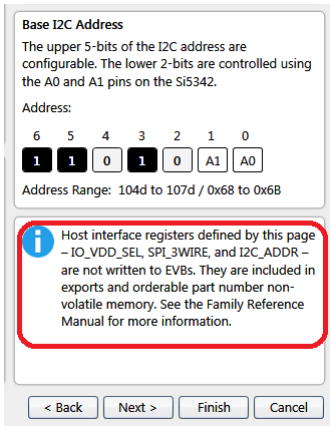

Figure 7.16. Base I2C Address

To permanently change the I2C base address on your device, you need to use the I2C Address Burn Tool. See the figures below to use the tool.

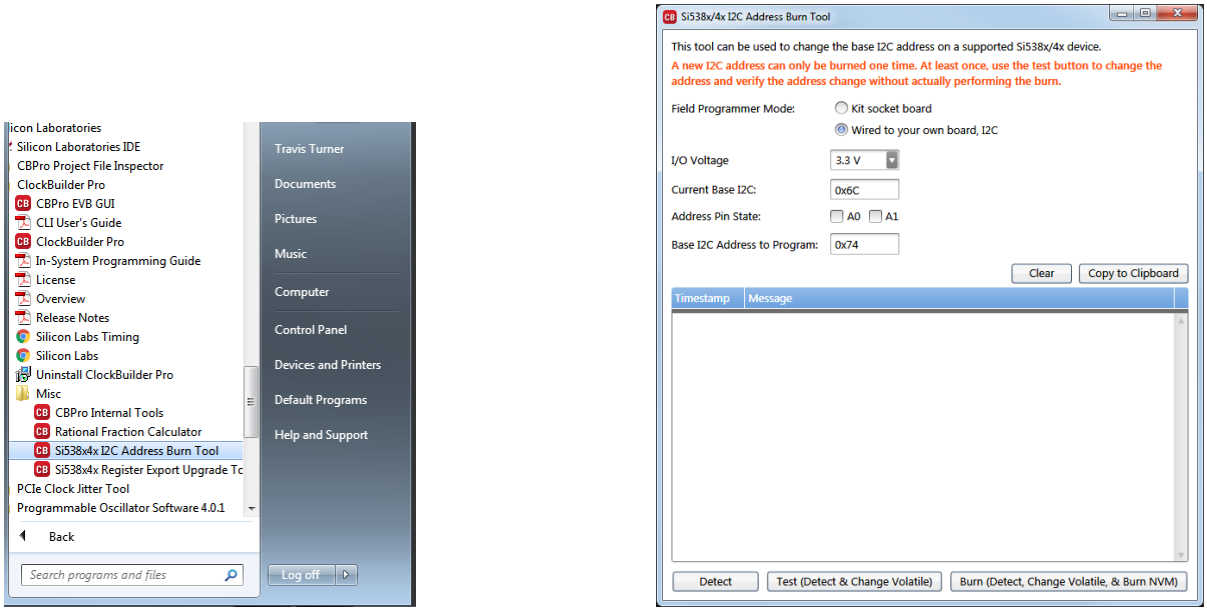

Figure 7.17. I2C Address Burn Tool

ClockBuilder Pro

One-click access to Timing tools, documentation, software, source code libraries & more. Available for Windows and iOS (CBGo only).

www.silabs.com/CBPro

Timing Portfolio
www.silabs.com/timing

SW/HW
www.silabs.com/CBPro

Quality
www.silabs.com/quality

Support and Community
community.silabs.com

SILICON LABS

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

http://www.silabs.com