AN200210

Getting Started with EZ-PD™ CCG3

Author: Vihang Trivedi

Associated Part Family: CYPD3xxx

Associated Software: EZ-PD™ Configuration Utility

Related Application Notes: AN210403

AN200210 introduces the USB Type-C EZ-PD™ CCG3 controller. It provides a brief overview of the CCG3 architecture and its features and applications and covers the evaluation kit in detail along with the development and debugging tools that can be used. It also references CCG3 resources to help you ramp up quickly with your product designs.

Contents

1 Introduction

EZ-PD CCG3 belongs to Cypress's family of USB Type-C microcontrollers that complies with the latest USB Type-C and Power Delivery (PD) standards. It consists of a dual bank of 64 KB of flash memory, 8 KB of SRAM memory, a crypto engine for authentication, and a pair of VCONN field-effect transistors (FETs). In addition, with the integrated Billboard controller, over-voltage protection (OVP), and over-current protection (OCP), it helps to reduce the need for additional components and the overall cost of a Type-C ecosystem. Typical applications using CCG3 include power adapters, power banks, Type-C dongles, notebooks, Thunderbolt hosts and cables, monitors, and docks.

1.1 EZ-PD CCG3 Features

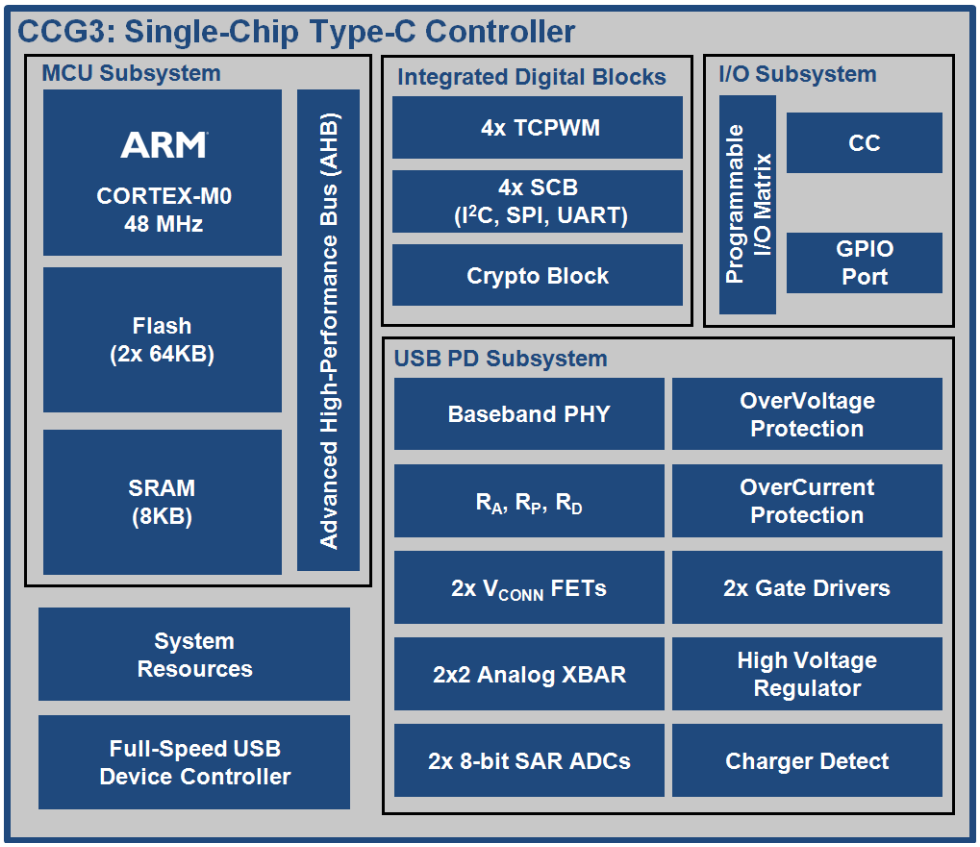
- 32-bit MCU subsystem
  - 48-MHz ARM® Cortex®-M0 processor
  - Dual 64-KB flash memory with fail-safe firmware updates

- USB Type-C support with PD
  - Integrated transceiver, supporting one Type-C port
  - Integrated upstream facing port—UFP (R_D), downstream facing port—DFP (R_P), and electronically marked cable assembly—EMCA (R_A) termination resistors
  - Compliance with USB PD, supporting all standard power profiles
- Integrated digital blocks
  - Four timers, counters, and pulse-width modulators (PWMs) and up to 20 general-purpose I/Os (GPIOs)
  - Four serial communication blocks (SCBs) for configurable master/slave I^2C, SPI, or UART
  - Integrated USB Billboard controller with Billboard Device Class support
  - Integrated crypto engine with USB authentication support
- Integrated analog blocks
  - 20-V operation
  - Two V_{BUS} gate drivers
  - 2x2 Crossbar switch

1.2 CCG3 Block Diagram

Figure 1 shows a block diagram of the CCG3 architecture. For more details, refer to the CCG3 datasheet.

Figure 1. CCG3 Architecture Block Diagram



1.3 Prerequisites

This section lists the hardware and software required to get started with CCG3 devices.

1.3.1 Hardware

- CY4531 EZ-PD CCG3 Evaluation Kit (EVK)
- PC/laptop with Windows 7 or later platforms; PC with at least one USB port (USB 3.0 port) and Windows 7 or later recommended
- Type-C power adapter that supports a 9-V or higher power profile

  Note: The CCG3 device works with all Type-C power adapters that support power profiles from 5 V to 20 V. This power profile requirement relates to the CY4531 CCG3 EVK architecture.
- CY4500 EZ-PD™ Protocol Analyzer (optional; required only for firmware debugging)
- MiniProg3 device (optional; required only to program the CCG3 device with a file in .hex format)

1.3.2 Software

- EZ-PD CCGx Software Development Kit (SDK) (Version 2.2 or later)
- EZ-PD Configuration Utility (version 1.00 or later)
- PSoC Creator™ 3.3 SP2 or later with PSoC Programmer 3.24.2 or later (required only if modifying base firmware functionality)

1.4 CCG3 Design Flow

This section describes a typical design flow users would go through during the Type-C application design right from the conceptual stage to manufacturing using CCG3 devices. It also covers how each of the hardware, software, and firmware resources described in this Application Note is used through the design flow. Figure 2 shows a typical design flow using CCG3 devices.

Once the CCG3 based Type-C application is determined and reference designs have been reviewed, the hardware and the application development phase of the design flow can be started in parallel. The hardware flow includes building reference schematics based on the end application and doing the board design to get a few prototypes ready for the next phase. These reference schematics can be based on the reference designs available on the Cypress' CCG3 webpage. The CY4531 EZ-PD CCG3 Evaluation Kit (EVK) design can also be reused especially for Type-C notebook based designs.

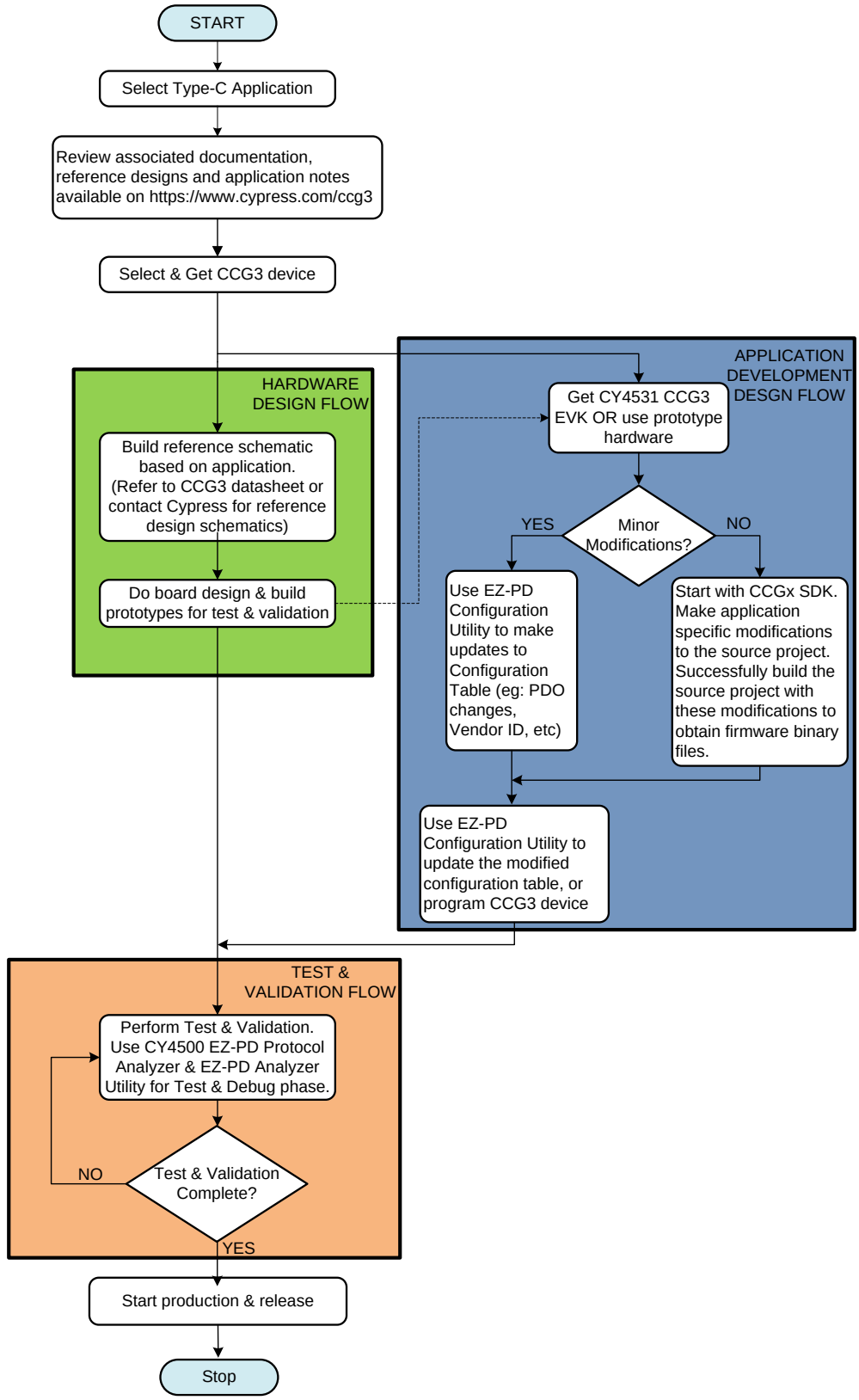
The application development flow can get started using the CY4531 EZ-PD CCG3 Evaluation Kit (EVK) in parallel. The EZ-PD Configuration Utility can be used to make minor updates to the configuration table of the CCG3 device (for example, Changing PDOs and Vendor ID changes). For making application specific modifications, users can use the EZ-PD CCGx Software Development Kit (SDK) (Version 2.2 or later).

This EZ-PD CCGx SDK (referred to as CCGx SDK through the rest of the document) along with PSoC® Creator™ (version 3.3 SP2 or later) can be used to build the source projects and create firmware binary files.

Once either configuration changes or firmware changes are made, the EZ-PD Configuration Utility can be used to update the modified configuration table or to program the CCG3 device. More information on which tool can be used for what purpose is covered in detail in Table 2.

Once the hardware and application development flow are completed, the existing system design is ready for the test and validation cycle. The CY4500 EZ-PD™ Protocol Analyzer can be used for testing, firmware debugging, and doing performance analysis. Mass production and manufacturing can start once the test and validation flow is over and the system design is final.

Figure 2. CCG3 Design Flow



1.5 CCG3 Resources

Table 1 lists the web resources available to help you design end applications using CCG3 devices.

Table 1. CCG3 Design Resources

Category	Available Resources
Datasheet	CCG3 datasheet
Hardware	CY4531 CCG3 EVK – Contains documentation and design files
Application Notes	AN210403 – Hardware Design Guidelines for Dual Role Port Applications Using EZ-PD USB Type-C Controllers
Programming Specifications Document	Programming specifications – Provides guidelines on how to program the flash memory of CCG3 devices
Host PC Software	EZ-PD CCGx SDK
	EZ-PD Configuration Utility – GUI-based Windows application to help you configure CCGx controllers
	PSoC Creator 3.3 SP2 or later (firmware development tool)
	PSoC Programmer 3.24.2 or later (firmware programming tool)
Debugging Tools	CY4500 EZ-PD™ Protocol Analyzer – Includes EZ-PD Analyzer Utility and documentation
Videos	USB Type-C Essentials, USB Type-C 101 Video Training Series
Other Collateral	CCG3 specific Knowledge Base Articles

Note: A basic introduction to Type-C and how to use Cypress' Type-C products can be found in the USB Type-C 101 Video Training Series of short videos (also listed above).

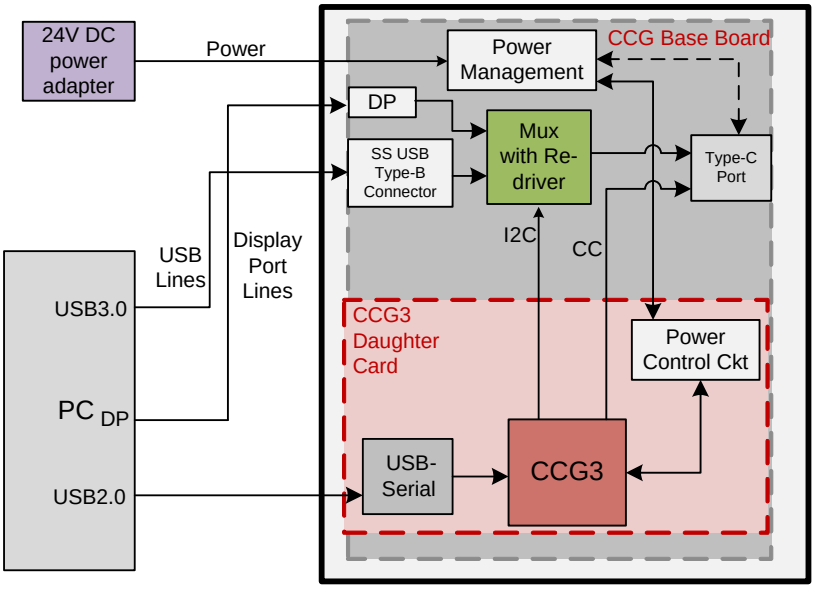
2 CCG3 Hardware Details

This section discusses the hardware to be used for getting started with the CCG3 device family. It focuses on the CY4531 EZ-PD CCG3 EVK and the CY4500 EZ-PD Protocol Analyzer.

2.1 CY4531 EZ-PD CCG3 EVK

The CY4531 EZ-PD CCG3 EVK consists of a base board and a daughter card. The CCG3 device is mounted on the daughter card, which is connected to the base board to enable the CCG3 device's Type-C port functionality. Figure 3 shows a block diagram of the CY4531 EZ-PD CCG3 EVK architecture.

Figure 3. CY4531 EZ-PD CCG3 EVK Architecture Block Diagram



DP – DisplayPort
PS – Power Supply
USB2 – Used only for programming

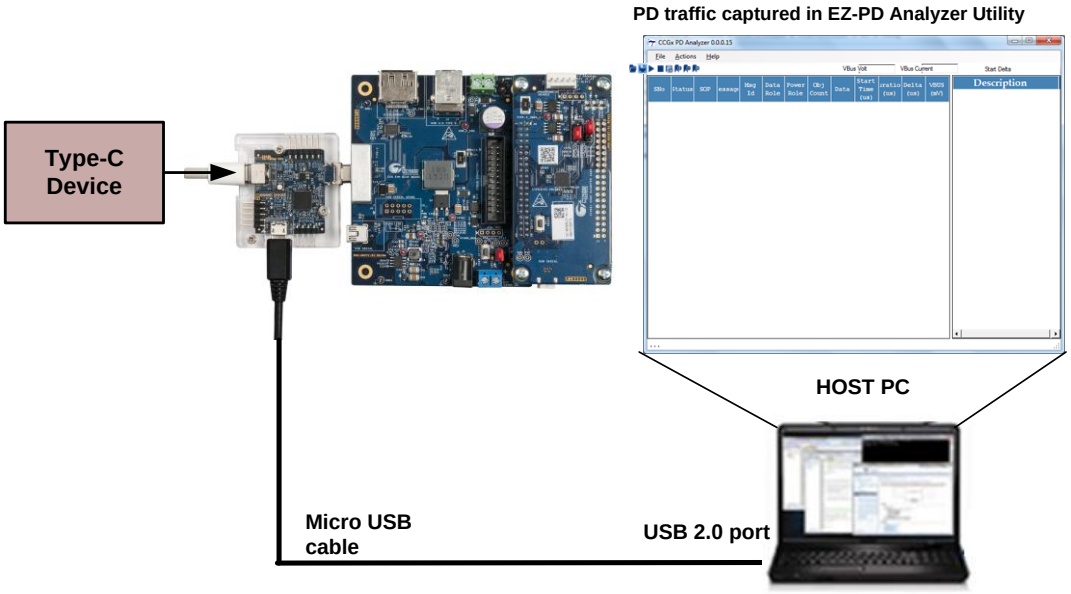
The CCG base board consists of a DC input power supply, display multiplexer, DisplayPort, SuperSpeed Type-B port, and Type-C port. The CCG3 daughter card consists of the CCG3 device and a USB-Serial IC to provide a USB interface for debugging and programming. The configuration channel (CC) lines of the CCG3 device are connected to the Type-C port. The display multiplexer is controlled by the CCG3 device over an I^2C interface.

The CY4531 EZ-PD CCG3 EVK includes power provider and consumer path control circuitry on the CCG3 daughter card to showcase EZ-PD CCG3's ability to switch its power role from a provider to a consumer and vice versa. It contains over-voltage protection (OVP) and over-current protection (OCP) circuitry for VBUS and supports EZ-PD CCG3 device programming over SWD and I^2C interfaces. The EVK supports PCs, notebooks, tablets, and other applications that would host a Type-C interface. It is primarily intended as an evaluation vehicle for USB Host systems that house a Type-C connector. Refer to the CY4531 EZ-PD CCG3 EVK Guide for in-depth information about EVK use cases and a Type-C notebook application using the EVK.

2.2 CY4500 EZ-PD Protocol Analyzer

Cypress's CY4500 EZ-PD Protocol Analyzer supports protocol analysis of the USB PD and USB Type-C specifications. It performs nonintrusive probing, and captures accurate protocol messages on the CC line. This analyzer consists of Cypress's programmable MCU (PSoC 5LP), which monitors the data on the CC line and sends the data to the host application over a USB interface. The Type-C plug and Type-C receptacle on this analyzer provide a pass-through for the Power Delivery (PD) packets transmitted between each Type-C PD connection. The processor MCU taps these PD packets without disturbing the system and transfers them over the USB interface to a PC running the host application. Figure 4 shows the connections between the CY4500 EZ-PD Protocol Analyzer and the CY4531 CCG3 EVK.

Figure 4. Connections between CY4500 EZ-PD Protocol Analyzer and CY4531 CCG3 EVK



Downloading and installing the latest analyzer software setup (*CY4500Setup.exe*) from the Cypress website http://www.cypress.com/CY4500 also installs the EZ-PD Analyzer Utility for running the analyzer, required drivers, and all relevant documentation. An overview of this utility is provided in the EZ-PD Analyzer Utility section.

3 CCG3 Firmware Details and Build Environment

This section provides an overview of the firmware architecture, organization, and operation, and introduces PSoC Creator and the CCGx SDK which are used as the build environment for firmware development. This section is followed by a section, which has an overview of the software tools that can be used for development and debugging. Before covering the in-depth details regarding the firmware build environment and software tools, a brief summary of which tool can be used for what purpose is provided in Table 2 below. Note that many configuration settings can be changed without the need for modifying the base firmware using the SDK. For a discussion of how the SDK fits into the CCG3 ecosystem, see the CCG3 Design Flow section.

Table 2. Firmware, Software Tools & Their Purposes

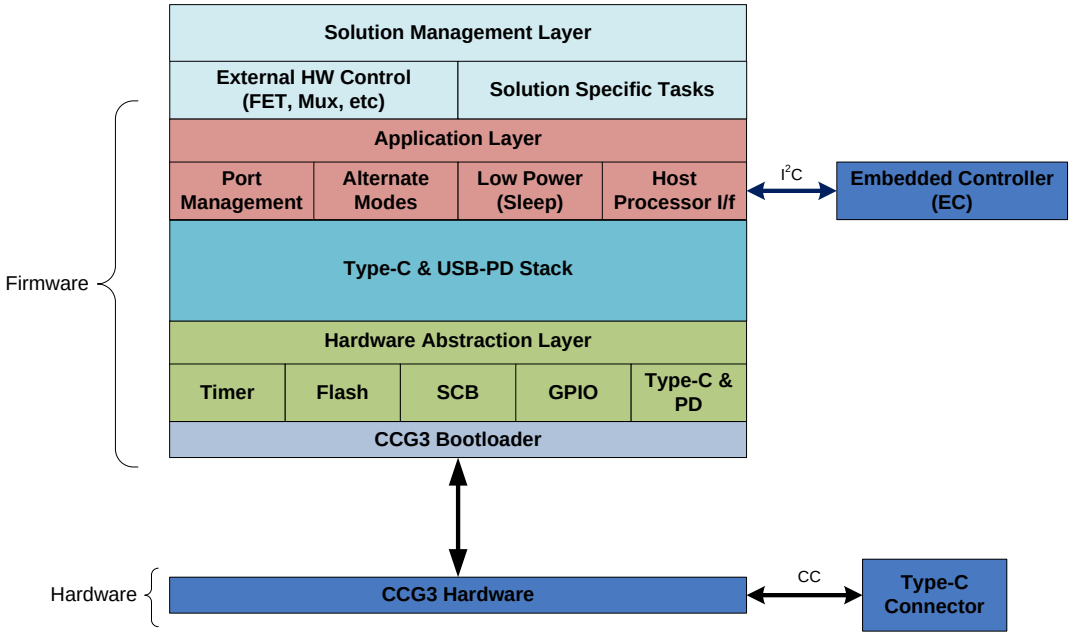
Tool	Purpose	Output and Its Use	Described in
CCGx SDK	It is the build environment for CCGx firmware development. It has application code with Type-C and USB-PD-compliant firmware, which can be used for further development.	Compiling the application source project generates firmware binaries, which can be used to program/reprogram respective CCGx devices.	CCGx SDK section
Firmware Configurable Features in CCGx SDK	Each application firmware in CCGx SDK contains a set of configurable features, which can be modified based on system requirements.	Compiling the application source project generates firmware binaries, which can be used to program/re-program respective CCGx devices.	Firmware Configurable Features section
EZ-PD Configuration Utility	Used for modifying parameters stored in the configuration table area of CCGx device flash. Also used for programming the CCGx device flash.	Configuration parameters can be read, modified, and saved for firmware development and debugging. Modified parameters can be updated in device flash intuitively without firmware development.	EZ-PD Configuration Utility section

3.1 CCG3 Firmware Architecture Overview

Figure 5 shows a block diagram of the CCG3 firmware architecture. The CCG3 firmware architecture allows users to implement a variety of USB-PD applications using the CCG3 device. It contains the following components:

- **Hardware Abstraction Layer (HAL):** This includes the low-level drivers for various hardware blocks of the CCG3 device. This includes drivers for the Type-C and USB-PD block, Serial Communication Blocks (SCBs), GPIOs, flash module, and timer module.
- **USB Type-C and USB-PD Protocol Stack:** This is the complete USB-PD protocol stack that includes the Type-C and USB-PD port managers, USB-PD protocol layer, the USB-PD policy engine, and the device policy manager. The device policy manager is designed to allow all policy decisions to be made at the application level, either on an external Embedded Controller (EC) or in the CCG3 firmware itself.
- **Application Layer:** This is the layer responsible for managing the functions of the PD port, handling alternate modes, power management, and also manages the Host Processor Interface (HPI). It is further sub-categorized into the following components:
  - **Port Management:** This module handles all of the PD port management functions including the algorithm for optimal contract negotiations, source and sink power control, source voltage selection, port role assignment, and swap request handling.
  - **Alternate Modes:** This module implements the alternate mode handling for CCG as a DFP and UFP. A fully tested implementation of the DisplayPort alternate mode with CCG as the DFP is provided. The module also allows users to implement their own alternate mode support in both DFP and UFP modes.
  - **Low Power:** This module attempts to keep the CCG device in the low power standby mode as often as possible to minimize power consumption.
  - **Host Processor Interface (HPI):** The Host Processor Interface (HPI) is an I^2C based control interface that allows an EC to monitor and control the runtime operation of the CCG3 device. CCG3 implements the HPI using an I^2C interface, with an interrupt line using a GPIO. In a typical CCG3 notebook application, the EC may communicate with the CCG3 device to negotiate the power with the connected Type-C device based on the charge level of the internal battery. CCG3 provides this functionality using commands, responses, events, and asynchronous messages that are modeled as registers. Detailed documentation on CCG3 HPI Interface is available. Contact your local Cypress FAE or contact Cypress Technical Support for this information.
- **Solution Management Layer:** This layer consists of the following components:
  - **External Hardware Control:** This is a hardware design dependent module that controls the external hardware blocks such as FETs, regulators, and Type-C switches.
  - **Solution specific tasks:** This is an application layer module where any custom tasks required by the user solution can be implemented.

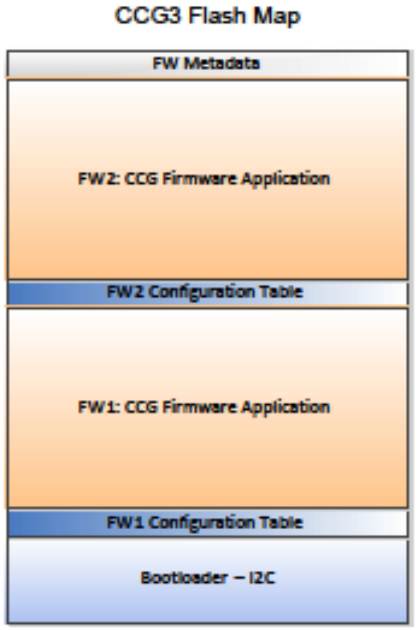
Figure 5. CCG3 Firmware Architecture Diagram



3.2 Flash Memory Organization

The CCG3 device has 128 KB of flash divided into two banks of 64 KB each, which allows support for dual firmware images. Dual firmware images enable firmware updates to be fail-safe; that is, the firmware update does not interrupt the normal operation of the device. All applications supported by CCG3 support dual firmware images. Figure 6 shows the flash map of the CCG3 device.

Figure 6. CCG3 Firmware Organization



The I²C bootloader is used to upgrade the CCG3 application firmware. It is allocated in a fixed area. The bootloader memory area can only be written using the SWD interface. This I²C bootloader uses 5kB of memory. The configuration table holds the PD configuration for the CCG3 application, and is located at the beginning of each firmware binary. The size of each configuration table is 1kB. A lot of configurations can be updated by modifying this table using the EZ-PD Configuration Utility. An overview of this utility is provided in the EZ-PD Configuration Utility section. The CCG firmware areas FW1 and FW2 are used for the CCG3 firmware application. The metadata area holds the metadata about both firmware binaries. The firmware metadata follows the definition provided by the PSoC Creator bootloader component, and includes the firmware checksum, firmware size, and the start address.

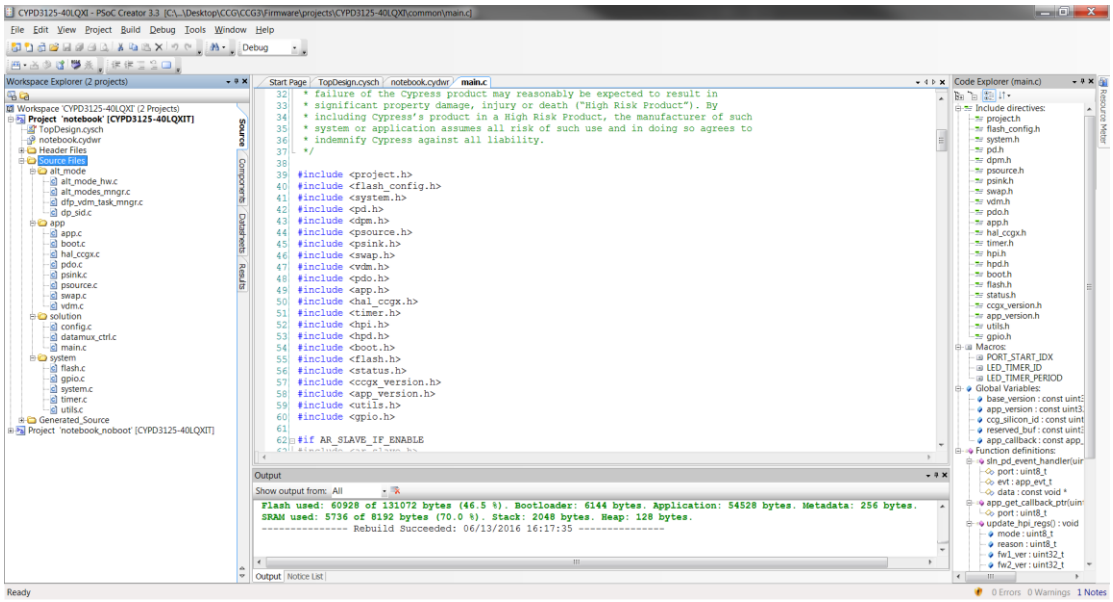
3.3 Firmware Build Environment

The tool used for CCG3 firmware development is PSoC Creator, which is a free, Windows-based Integrated Design Environment (IDE). PSoC Creator 3.3 SP2 or later and the CCGx SDK are required to edit, compile, download, and debug the firmware for the CCG3 Notebook application, as shown in Figure 7. The PSoC Creator compiler tool chain is ARM GCC (build 493, provided along with the PSoC Creator build). Refer to the PSoC Creator User Guide for more details on the PSoC Creator build environment.

Visit the PSoC Creator product page to download and install the latest version of PSoC Creator (3.3 SP2 or later). This web page also contains links to video training and additional documentation. Within the PSoC Creator tool, additional help is available via the following documents:

- **Quick Start Guide:** Choose **Help** > **Documentation** > **Quick Start Guide**. This guide gives you the basics for developing PSoC Creator projects.
- **System Reference Guide:** Choose **Help** > **System Reference Guide**. This guide lists and describes the system functions provided by PSoC Creator.
- **Document Manager:** PSoC Creator provides a Document Manager to help you easily find and review document resources. To open the Document Manager, choose **Help** > **Document Manager**.

Figure 7. PSoC Creator IDE



3.3.1 CCGx SDK

The CCGx Software Development Kit (SDK) is a PSoC Creator project that allows users to harness the capabilities of Cypress's CCG families of Type-C Controllers. It provides a Type-C and USB-PD specification compliant firmware stack along with the necessary drivers and software interfaces required to implement applications using CCG controllers. The CCGx SDK also includes reference projects implementing standard Type-C applications and documentation that guides users in customizing existing applications, or creating new ones. For more information on the CCGx SDK, refer to the CCGx SDK User Guide.

This Application Note references only the notebook application for CCG3 devices. In general, the CCGx SDK consists of the following basic components.

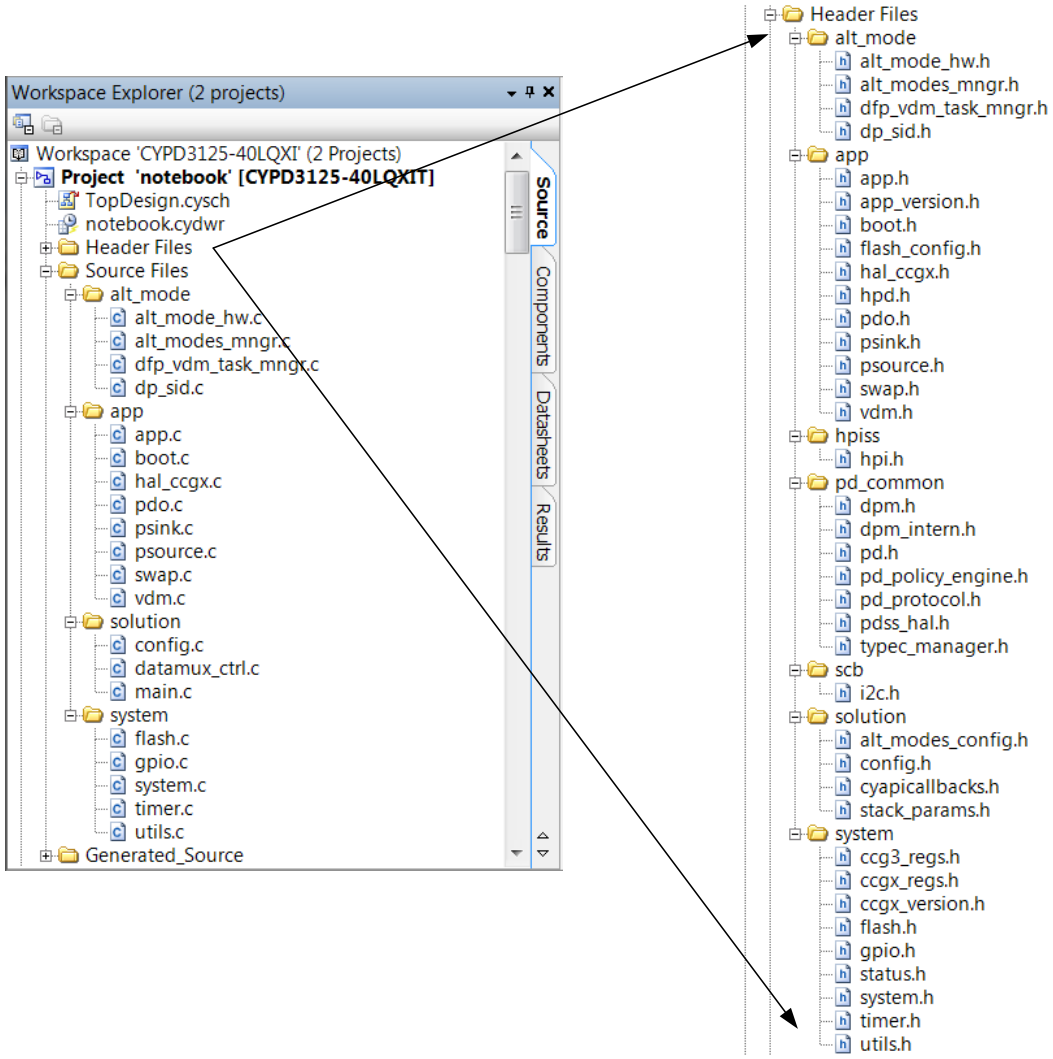
- Source Code
  - PD Stack and Host Processor Interface (HPI) in a pre-compiled library form
  - Firmware sources for other blocks
  - Reference Application
  - Integrated UFP (R_D), DFP (R_P), and EMCA (R_A) termination resistors
  - Compliance with USB PD, supporting all standard power profiles
- Firmware Binaries
  - Application specific *.hex* and *.cyacd* files (this application note references the CCG3 notebook application only)
- Supporting Documentation
  - CCGx SDK User Guide
  - Firmware API Reference Guide
  - Release Notes

3.3.2 PSoC Creator Project Structure Overview

The CCGx SDK includes a reference firmware project for a notebook application of the CCG3 device. Figure 8 shows the PSoC Creator workspace file structure for a CCG3 device–based notebook application. PSoC Creator generates bootloadable *.cyacd* files and a Cypress format *.hex* file every time a project is successfully compiled and built.

In every PSoC Creator project, a system-level design diagram is included. This schematic is located in the *TopDesign.cysch* file. The firmware version is controlled by an ‘application type’ string and ‘application major’ and ‘application minor’ numbers. This information is stored in the *app_version.h* file of the PSoC Creator project. The application string and the version numbers can be modified by customers based on the updates needed and requirements.

Figure 8. PSoC Creator Workspace Structure for CCG3 Notebook Application



3.4 Firmware Configurable Features

The notebook application firmware in the CCGx SDK has the following configurable features, which can be modified by individual users based on system requirements. These configurable parameters are located in the *config.h* file. Table 3 lists the compile-time configurable features available in the CCG3 notebook application firmware.

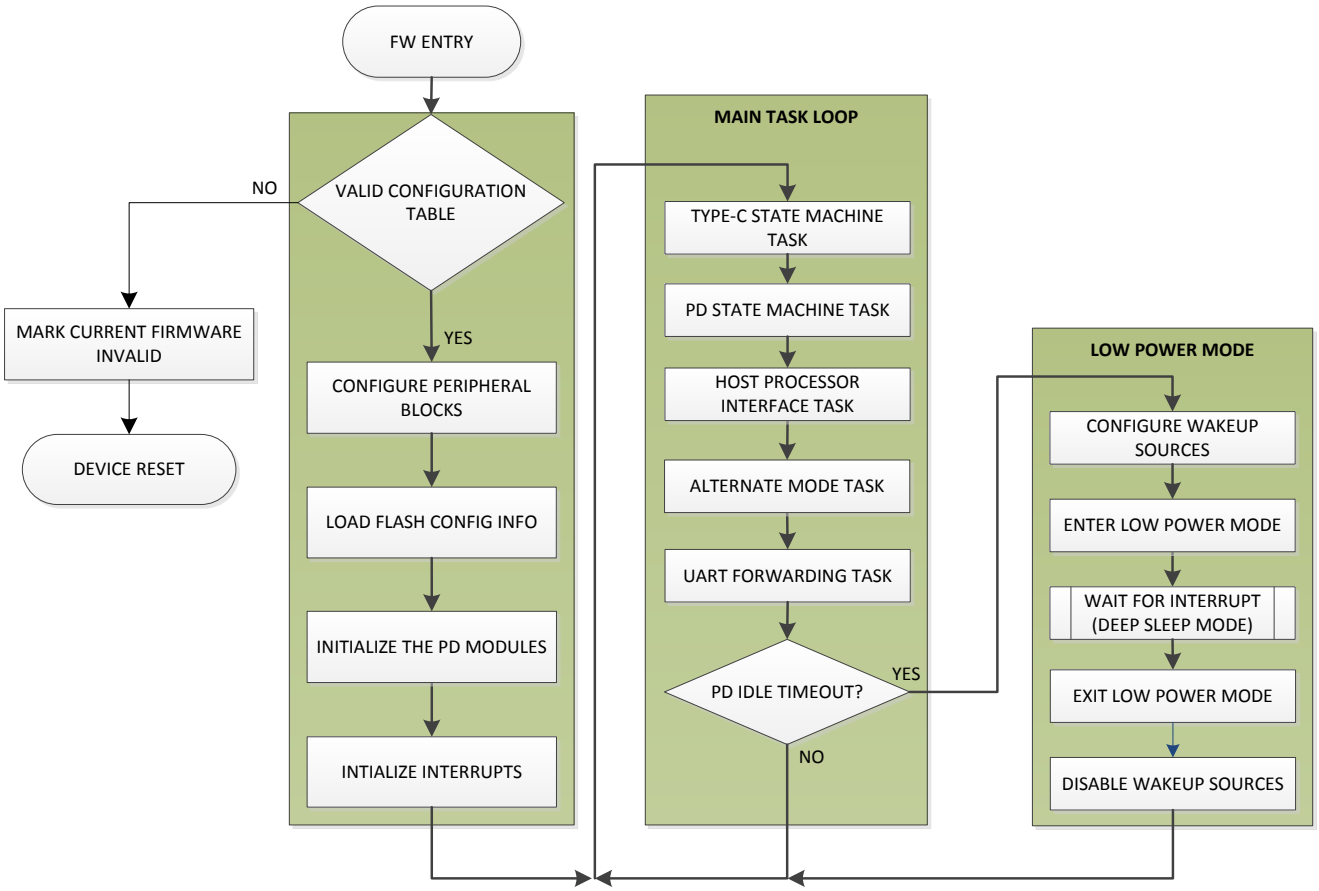
Table 3. Configurable Features of CCG3 Notebook Application Firmware (*config.h* file)

Pre-processor switch	Description	Values (Default values in bold)
VBUS_OVP_ENABLE	Enable overvoltage Protection handling on VBUS. This feature can be turned off using the configuration table, even if it is enabled here.	**1 for VBUS OVP enable** 0 for VBUS OVP disable
VBUS_OVP_AUTO_CONTROL_ENABLE	Enable automatic FET control by hardware when an OVP event is detected.	**1 for automatic hardware cut-off** 0 for firmware cut-off
SYS_DEEPSLEEP_ENABLE	Enable flag for the low power module, which keeps CCG3 in deep sleep mode at all possible times.	**1 for low power enable** 0 for low power disable
DFP_ALT_MODE_SUPP	Enable Alternate Mode handling when CCG3 is used as DFP.	**1 for alternate mode enable** 0 for alternate mode disable
DP_DFP_SUPP	Enable DisplayPort Alternate mode when CCG3 is used as DFP. This requires DFP_ALT_MODE_SUPP.	**1 for DisplayPort enable** 0 for DisplayPort disable
APP_FW_LED_ENABLE	Enable toggling of the LED to indicate firmware operation. It is recommended that this be left disabled in production designs to save power and also to free up the GPIO used for LED control.	**1 for LED enable** 0 for LED disable

3.5 Firmware Operation

The code flow for the application is implemented in the *Source Files\Solution\main.c* file. Figure 9 shows the flow diagram of the firmware operation. Based on this, it is clear that the firmware implementation is a simple round-robin loop that services each of the tasks that the application has to perform. All PD management, HPI command handling, and Vendor Defined Message (VDM) handling is encapsulated in the task handlers in the CCG3 firmware stack.

Figure 9. CCG3 Notebook Firmware Flow Diagram



More details on the CCGx SDK firmware architecture, firmware APIs, and getting started with the CCG3 SDK can be found in the CCGx SDK User Guide. The same guide can also be used for customizing the CCG3 firmware application, that is, modifying and updating the PSoC Creator project for a different hardware design, building and debugging the project, and programming the modified firmware in the CCG3 device.

3.6 Programming Firmware in CCG3 Devices

There are two methods to program CCG3 devices:

- Using EZ-PD Configuration Utility
- Using MiniProg3 device and PSoC Creator/PSoC Programmer tool

The EZ-PD Configuration Utility requires a *.cyacd* file as the input for the firmware binary file to be programmed. Refer to Chapter 4 of the CY4531 EZ-PD CCG3 Evaluation Kit User Guide for more details on how to program CCG3 devices using the EZ-PD Configuration Utility. The PSoC Creator/PSoC Programmer tool uses a *.hex* file for programming all devices. Refer to the knowledge base article KBA96477 for more details on how to program CCG3 devices using PSoC Creator/PSoC Programmer along with a MiniProg3 device.

4 Software Tools for CCG3 Application Firmware Development and Debugging

This section provides an overview of the EZ-PD Configuration Utility and the EZ-PD Analyzer Utility. In general, the tools can be categorized as follows:

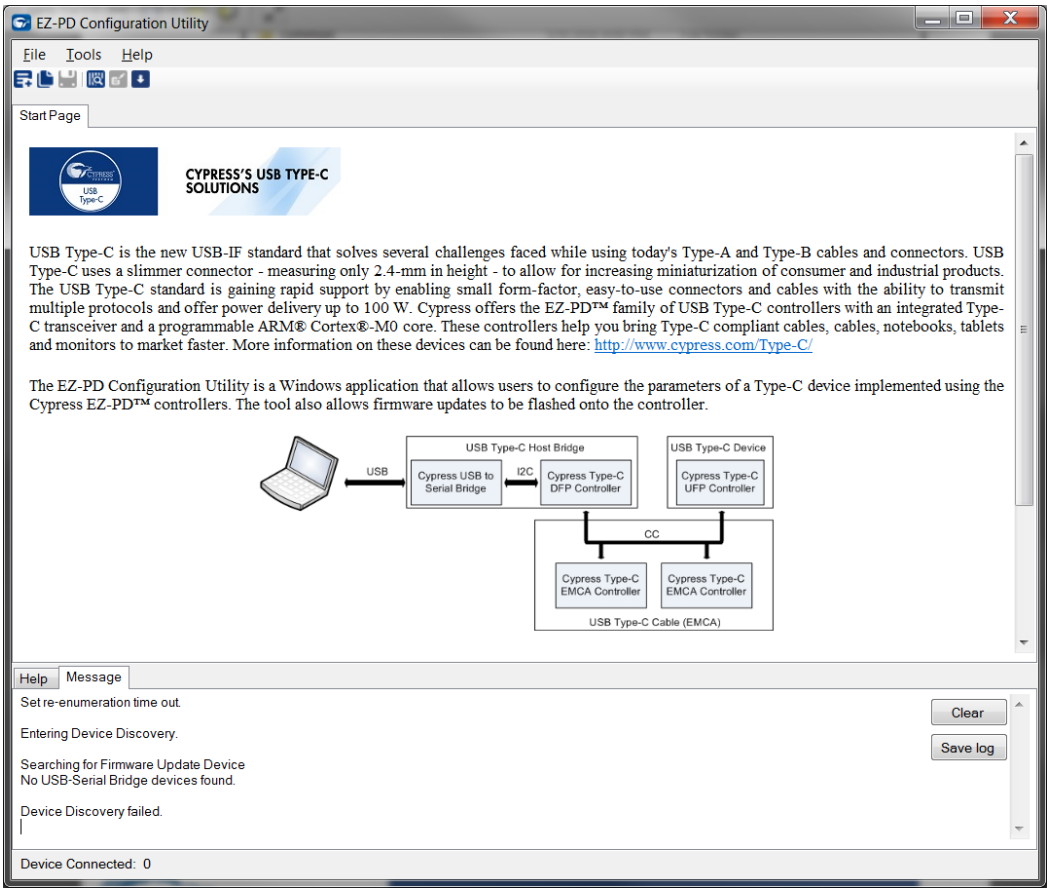
- **Firmware development and programming tools**
  - EZ-PD Configuration Utility: The EZ-PD Configuration Utility is used to read, modify, and update the configuration parameters of a CCGx device using the I^2C interface. It is also used to update the application firmware of the CCGx device.
  - PSoC Creator™ (CCG3 devices are supported from version 3.3 SP2 or later): As described in the Firmware Build Environment section, PSoC Creator is used to modify, debug, and program the firmware into the CCG3 device. This option is required only if the functionality of the CCG3 firmware is being modified from that of the standard firmware, or if the firmware is being modified for a different hardware design.
- **Debugging tool for CCGx applications**
  - EZ-PD Analyzer Utility: The EZ-PD Analyzer Utility along with a CY4500 EZ-PD Protocol Analyzer is used to capture PD messages between the CCG3 device and an attached Type-C device with the firmware application that is running. An overview of its functionality is provided in the CY4500 EZ-PD Protocol Analyzer section.

4.1 EZ-PD Configuration Utility

The EZ-PD Configuration Utility is a Windows application that configures the parameters stored in the configuration table areas of the internal flash memory of the CCGx device. These parameters can be chosen based on customer-specific application or system requirements. The utility allows you to intuitively select and configure the parameters for your specific application and thus saves time on firmware development. This utility also allows programming of the firmware applications.

You can download and install this tool from the Cypress website. After installation, the utility can be executed from the following location: **Windows** > **Start** > **All Programs** > **Cypress** > **EZ-PD Configuration Utility** > **EZ-PD Configuration Utility**. Figure 10 shows the utility running on a Windows machine.

Figure 10. EZ-PD Configuration Utility Start Page

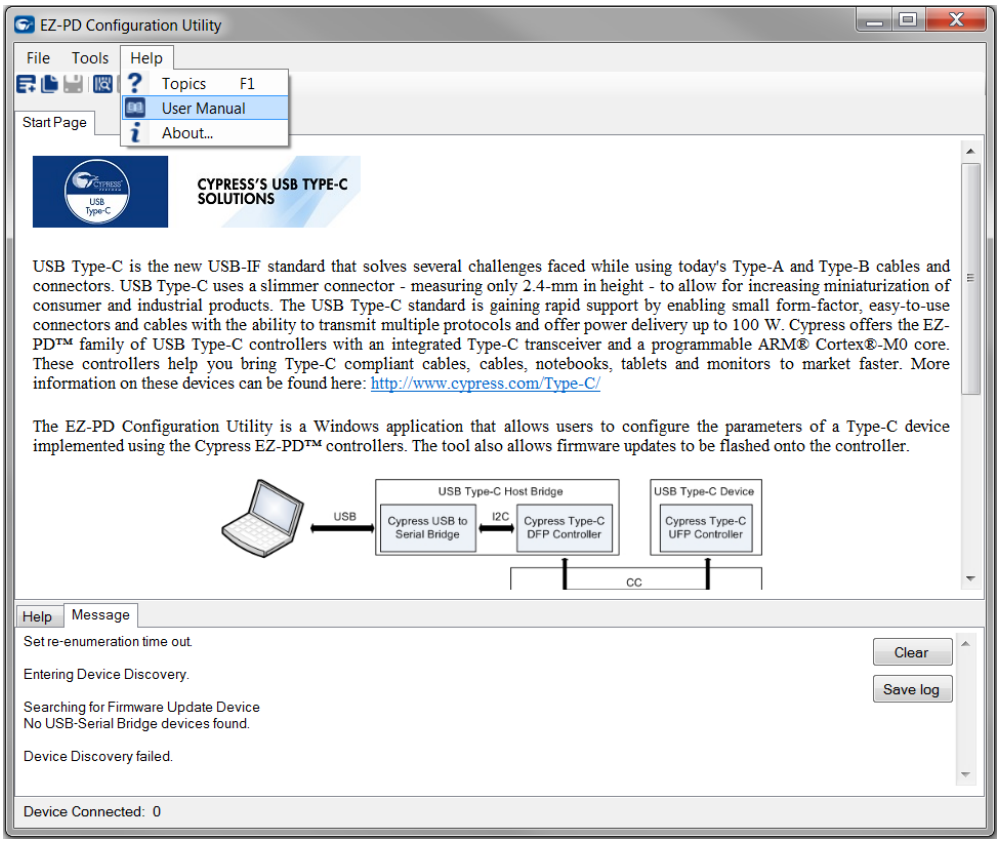


The EZ-PD Configuration Utility can be used to update the application firmware and configure the CCGx device. The utility shows the target application for the CCG3 device as a dual role port (DRP) notebook. The workflow for configuring any CCGx device is completed in three stages:

1. **Create configuration:** Create a new configuration from the **File** menu of the utility, or read an existing configuration to be modified.
2. **Select parameters:** Select the parameters available for the target application such as a notebook.
3. **Device configuration:** Program the device flash using the **Configure Device** option.

Refer to the EZ-PD Configuration Utility User Manual for more details on firmware update and configuration of the device. You can open the user manual by choosing **Help** > **User Manual** in the EZ-PD Configuration Utility, as shown in Figure 11.

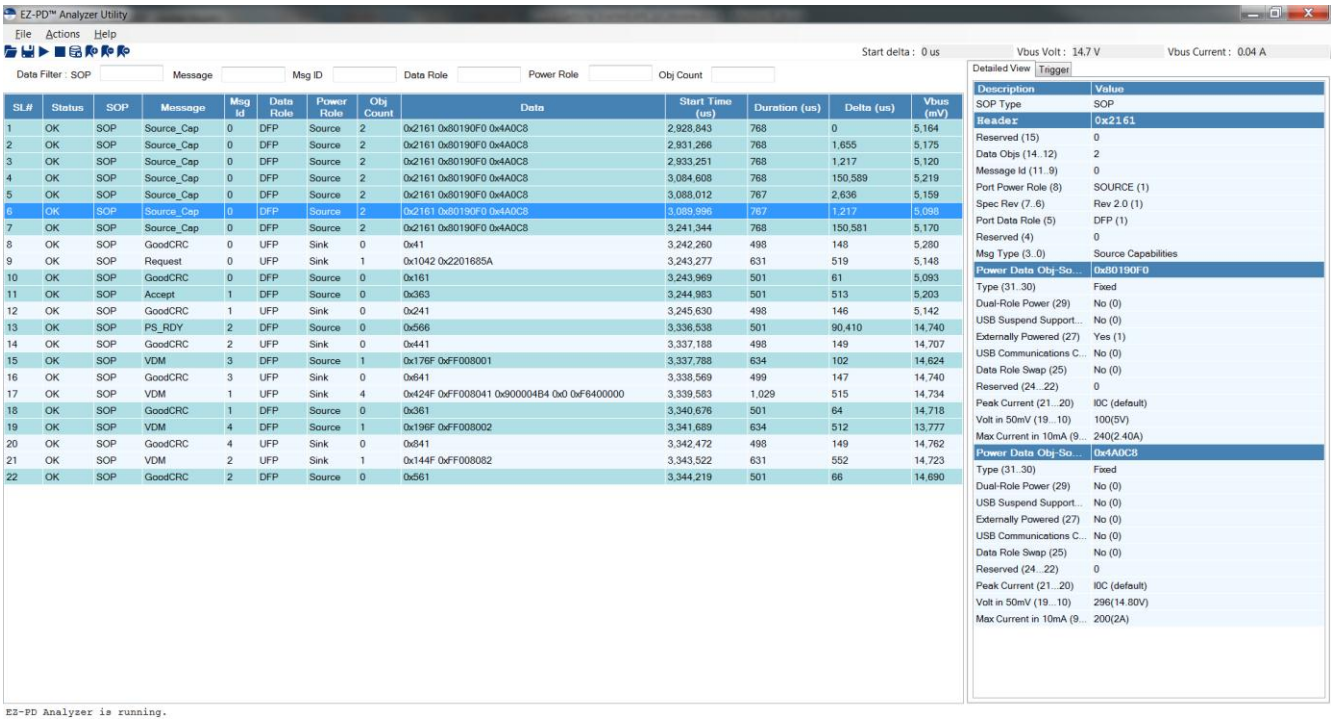
Figure 11. Opening the EZ-PD Configuration Utility User Manual



4.2 EZ-PD Analyzer Utility

The EZ-PD Analyzer Utility is a software application that is run along with a CY4500 EZ-PD Protocol Analyzer. When this utility is run, it shows PD messages over CC (for example, PR_SWAP, DR_SWAP, and PDOs) as shown in Figure 12 while the CCG3 device is establishing a PD contract with the connected Type-C device.

Figure 12. EZ-PD Analyzer Utility Showing PD Packets Captured on CC Line



Whenever a Type-C device is connected to the Type-C port of the CY4531 CCG3 EVK, a PD contract is established between the CCG3 device and the attached Type-C device. The flow of CC messages as well as source and sink Power Data Objects (PDOs—used to expose source/sink port's power capabilities in the CC messages) between the power port partners can be monitored on the PC by using the EZ-PD Protocol Analyzer and running the EZ-PD Analyzer Utility. This information helps in debugging the CCG3 firmware if there are any issues (errors, delays, no contract, and so on) related to establishing the power contract between the CCG3 device and the attached Type-C device.

5 CCG3 Configuration Parameter Modification Example

Many CCG configuration parameters are stored in the configuration tables in the device's internal flash memory. You may need to make changes to these parameters based on system requirements and the end application. This section provides a simple example of a Type-C ecosystem to describe how to change the configuration parameters using the EZ-PD Configuration Utility and how to verify the changes using the CY4500 EZ-PD™ Protocol Analyzer.

Consider the example of a Type-C power adapter connected to the CY4531 CCG3 EVK. Assuming this Type-C power adapter supports a custom PDO, the power contract between it and the CCG3 device of CY4531 CCG3 EVK will be successfully established only if the CCG3 device supports the same custom PDO. The EZ-PD Configuration Utility allows you to modify parameters such as the sink or source PDOs without making any changes to the base firmware so that the CY4531 CCG3 EVK can successfully establish a power contract with the power adapter.

From a top-level perspective, the following are the steps to modify the CCG3 device's configuration parameters. The following sections explain each step in detail.

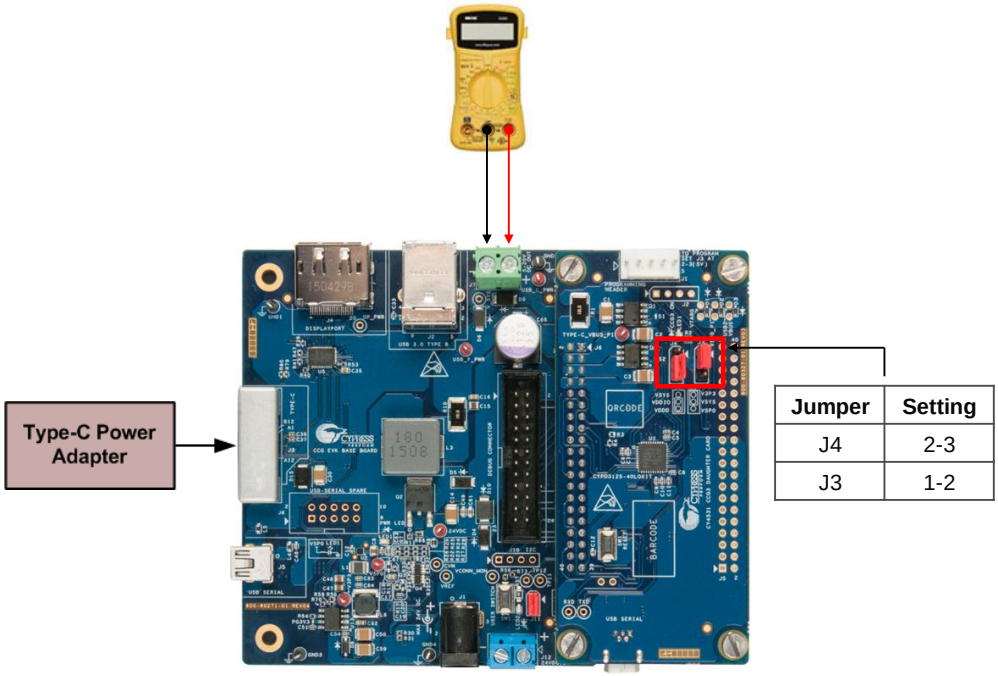
1. Test the CY4531 CCG3 EVK in the default configuration with a Type-C power adapter to identify the established power contract.
2. Update the configuration parameters (remove a sink PDO supported by the power adapter's custom PDOs such as 14.8V, 2A) using the EZ-PD Configuration Utility.
3. Program the updated configuration parameters into the CCG3 device using the EZ-PD Configuration Utility and verify.
4. Retest the CY4531 CCG3 EVK setup with a Type-C power adapter to verify the updated PDOs using the CY4500 EZ-PD Protocol Analyzer.

5.1 Test CY4531 CCG3 EVK Setup with the Default Configuration and Type-C Power Adapter

This section describes the steps to observe the behavior when the CY4531 CCG3 EVK with the default configuration parameters is connected to a Type-C power adapter. The voltage measured manually at the output header J7 is verified by using the CY4500 EZ-PD Protocol Analyzer as well.

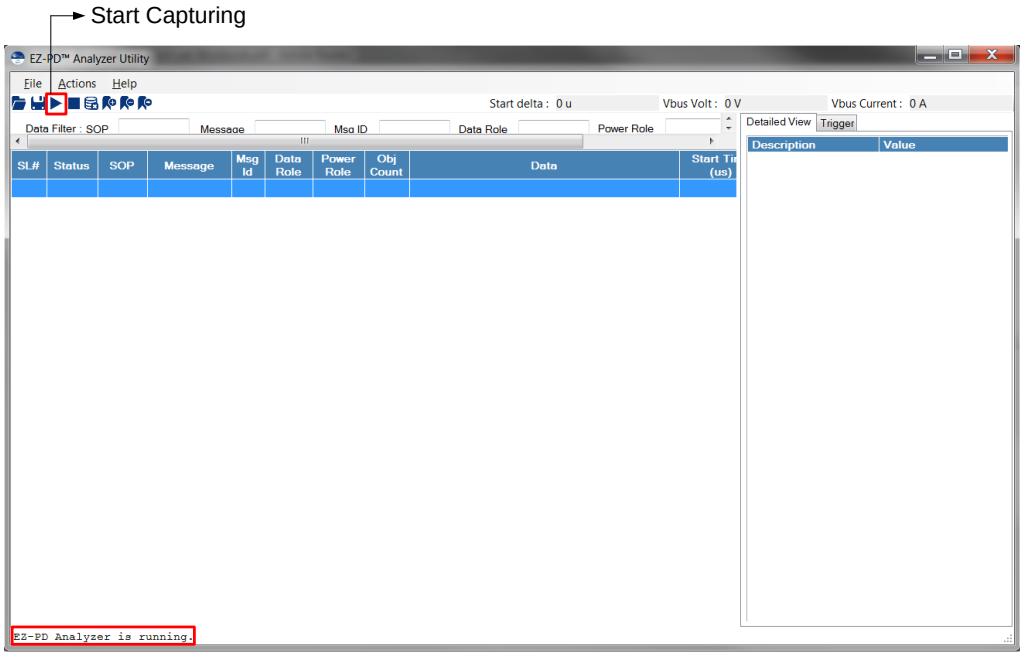
1. Ensure that the CCG3 device of the CY4531 CCG3 EVK is programmed with the latest default binaries provided on the CCGx SDK webpage. As shown in Figure 13, ensure jumper J4 is set to 2-3 and jumper J3 is set to 1-2 positions.
2. Connect the Type-C power adapter (Apple 29W power adapter used as reference in this example) to the CY4531 CCG3 EVK, as shown in Figure 13. Connect a multimeter to power output header J7 of the CY4531 CCG3 EVK.
3. When the Apple Type-C power adapter is connected to the Type-C port of the CY4531 CCG3 EVK, the CCG3 device of the CY4531 CCG3 EVK will be able to establish a power contract with the power adapter, as it is configured for custom PDOs supported by this Type-C power adapter. Due to this, the power adapter will provide 14.8V, which is the established power configuration. This can be verified by measuring the voltage on the power output header J7 of the CCG baseboard using a multimeter. The output voltage on this header will be about 14.8V, which is the negotiated voltage supported by the Type-C power adapter.

Figure 13. CY4531 CCG3 EVK Connected to Type-C Power Adapter



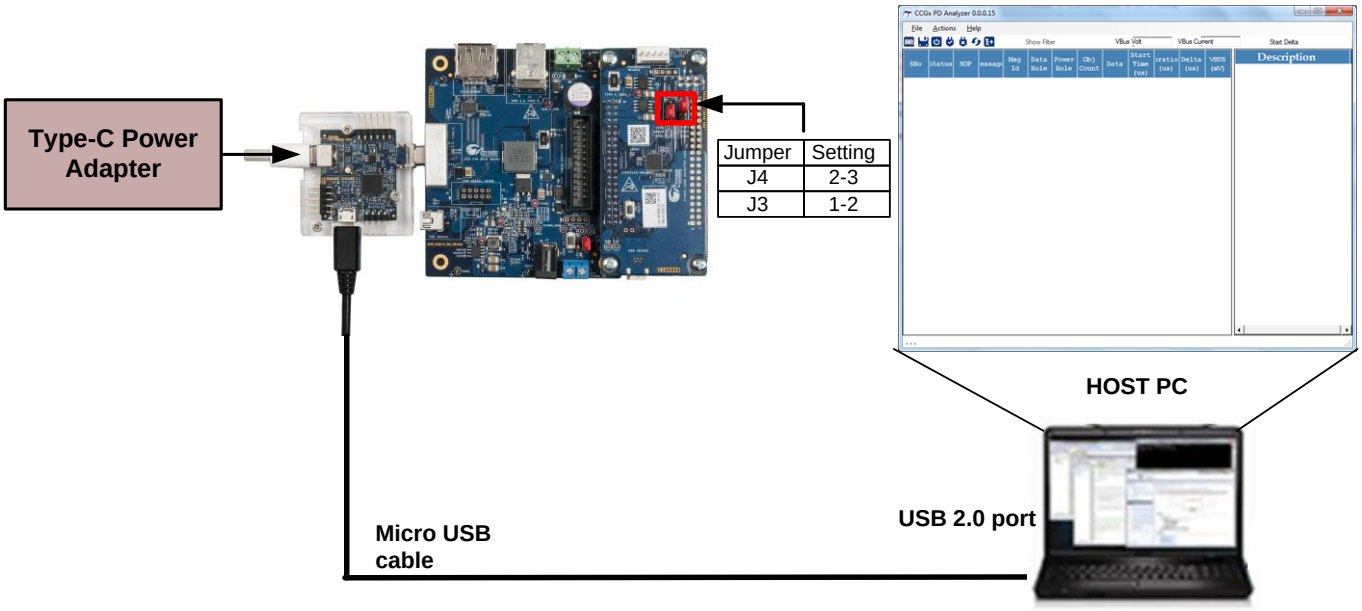
4. This output voltage can also be verified using the CY4500 EZ-PD Protocol Analyzer and capturing a CC trace. Ensure that the latest version of the CY4500 EZ-PD Protocol Analyzer is downloaded and installed.
5. Disconnect all connections shown in Figure 13, and connect CY4500 EZ-PD Protocol Analyzer to the PC (USB host) using a Micro-USB cable.
6. Connect the Type-C plug of the CY4500 EZ-PD Protocol Analyzer to the Type-C port of the CCG3 EVK.
7. Launch the EZ-PD Analyzer Utility from **Windows** > **All Programs** > **Cypress** > **EZ-PD Analyzer Utility** > **EZ-PD Analyzer Utility** and click on the **Start Capturing** icon shown in Figure 14 to start capturing the CC traffic.

Figure 14. Start Capturing Traffic in EZ-PD Analyzer Utility



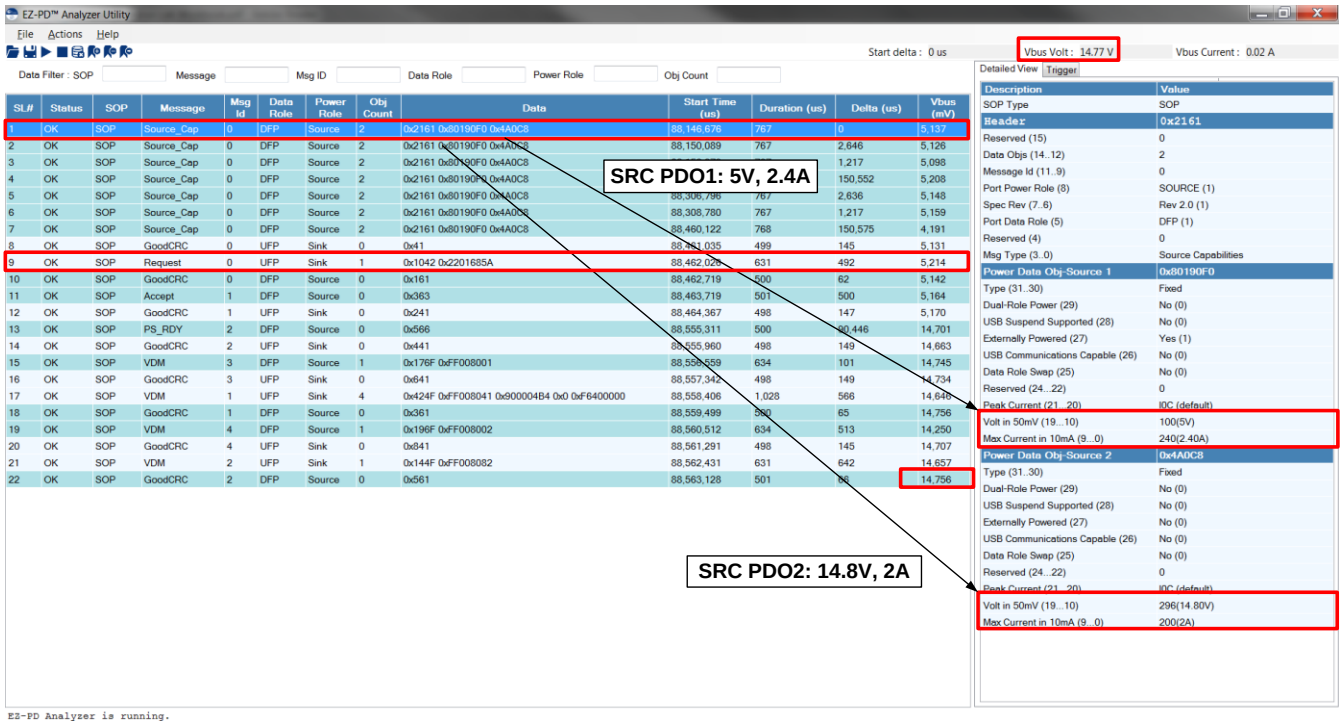
8. Connect the Apple 29W USB-C power adapter to the Type-C receptacle of the CY4500 EZ-PD Protocol Analyzer (see Figure 15). Observe that the PD traffic is being captured on the EZ-PD Analyzer Utility. The utility running on the host PC should look similar to that in Figure 16.

Figure 15. Connecting CY4500 EZ-PD Protocol Analyzer to CY4531 CCG3 EVK



9. A successful PD contract can be seen from the PD message sequence. From Figure 16, it is clear that the Type-C power adapter used in this example (Apple 29W USB-C power adapter) is a DFP (Source/Power Provider) and the Type-C port of the CY4531 CCG3 EVK is a UFP (Sink/Power Consumer). Looking at the 'Vbus' column in Figure 16, it can be seen that the V_{BUS} voltage is set to 14.8V at the end of the power contract negotiation.

Figure 16. EZ-PD Analyzer Utility Showing PD Packets Captured on CC Line



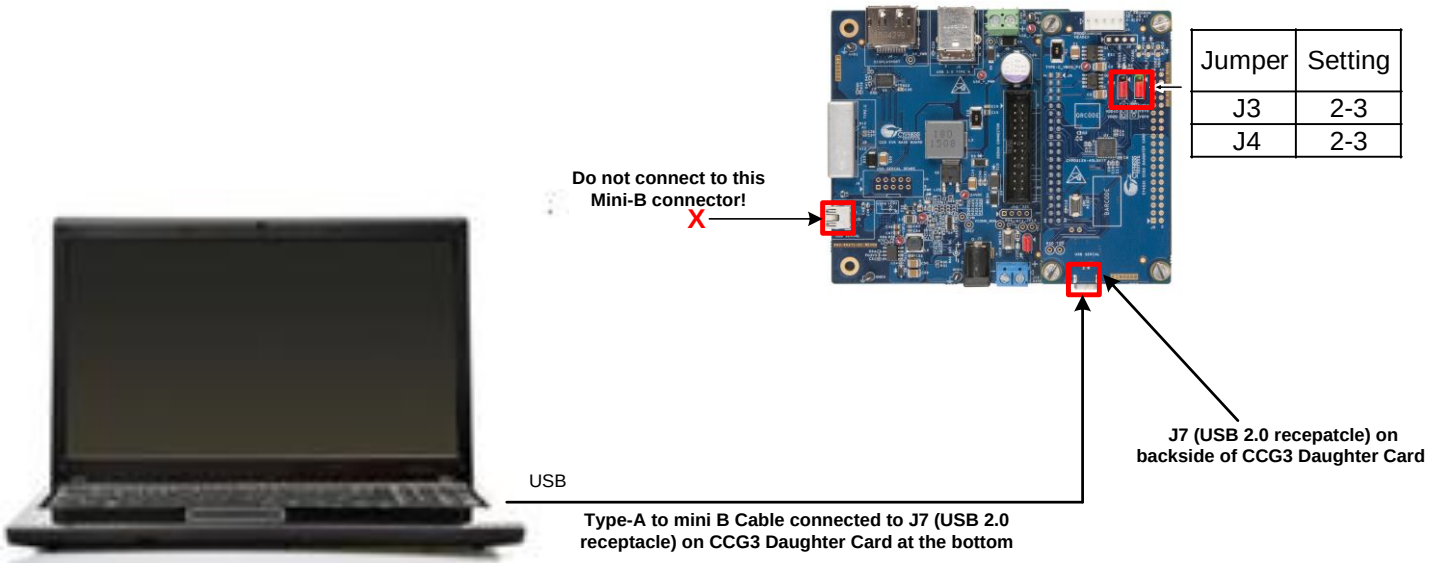
10. The CC trace capture shown in Figure 16 shows that a power contract for 14.8 V is established, and the Type-C power adapter is providing 14.8 V to the device. During the power negotiation phase, the power provider (that is, the Type-C power adapter) sends the **Source_Capabilities** message to the attached power consumer device (that is, the CY4531 CCG3 EVK). The **Source_Capabilities** message in Figure 16 shows the custom PDOs (**Power Data Obj-Source 1** and **Power Data Obj-Source 2**) supported by the Type-C power adapter.
11. The **Request** message after the **Source_Capabilities** message is always sent by the power consumer device to request the power. Since the CCG3 device can establish a power contract for 14.8 V, it requests the PDO object at position 2, which corresponds to the 14.8V PDO. Refer to USB Power Delivery (PD) Spec Revision 2.0, Version 1.1 for more details on PD messages.
12. Looking at the 'Vbus' column in Figure 16 and as mentioned in step 9, it can be seen that the V_{BUS} voltage is set to 14.8V at the end of the power contract negotiation. This confirms that at the initial stage of the setup, the power contract of 14.8V is established between the Type-C power adapter and the CY4531 CCG3 EVK.

5.2 Modify Configuration Parameters Using EZ-PD Configuration Utility

This section covers the steps for updating the configuration parameters and programming the CCG3 device using the EZ-PD Configuration Utility. Ensure that the latest version of EZ-PD Configuration Utility is downloaded and installed for successfully executing the steps described in this section.

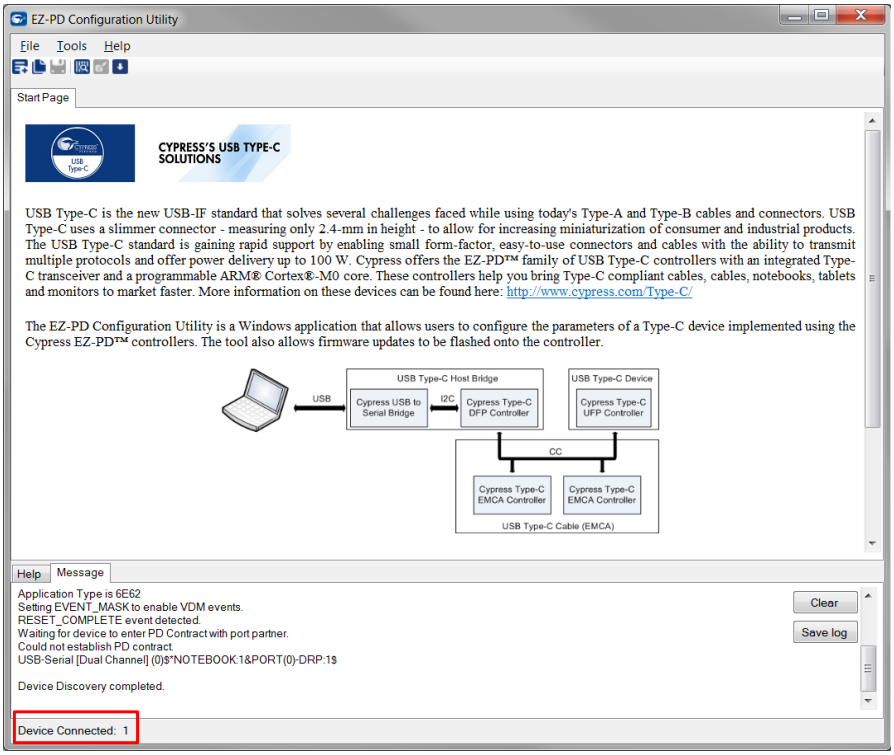
1. Disconnect the hardware setup described in Test CY4531 CCG3 EVK Setup with the Default Configuration and Type-C Power Adapter.
2. As shown in Figure 17, ensure jumper J4 and J3 of CY4531 CCG3 EVK's daughter card are set to the 2-3 positions. Connect the CY4531 CCG3 EVK setup to the PC using a USB 2.0 Type-A to Mini-B cable. The CY4531 CCG3 EVK's daughter card's USB2.0 receptacle at the J7 connector (on the backside of the CCG3 daughter card) must be used for this setup as shown in Figure 17. Wait for driver detection and binding for the USB-Serial controller on the CY4531 CCG3 EVK.

Figure 17. Setting Up the CY4531 CCG3 EVK to Use with EZ-PD Configuration Utility



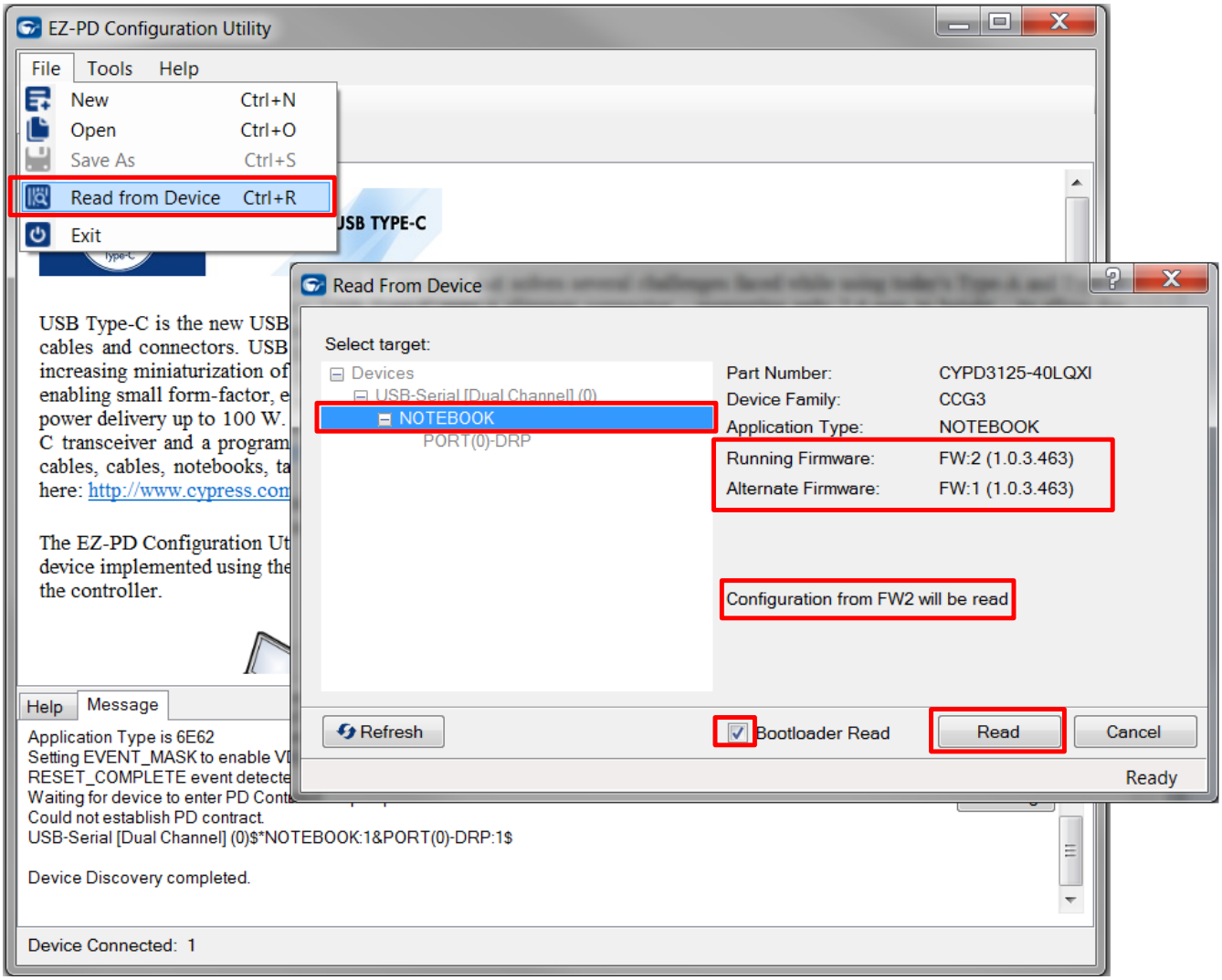
3. Launch the EZ-PD Configuration Utility from **Windows** > **All Programs** > **Cypress** > **EZ-PD Configuration Utility** > **EZ-PD Configuration Utility.** If the device driver binding is successful, the GUI should report one device connected on the status bar of the GUI as shown in Figure 18.

Figure 18. EZ-PD Configuration Utility Showing Successful Connection to CY4531 CCG3 EVK



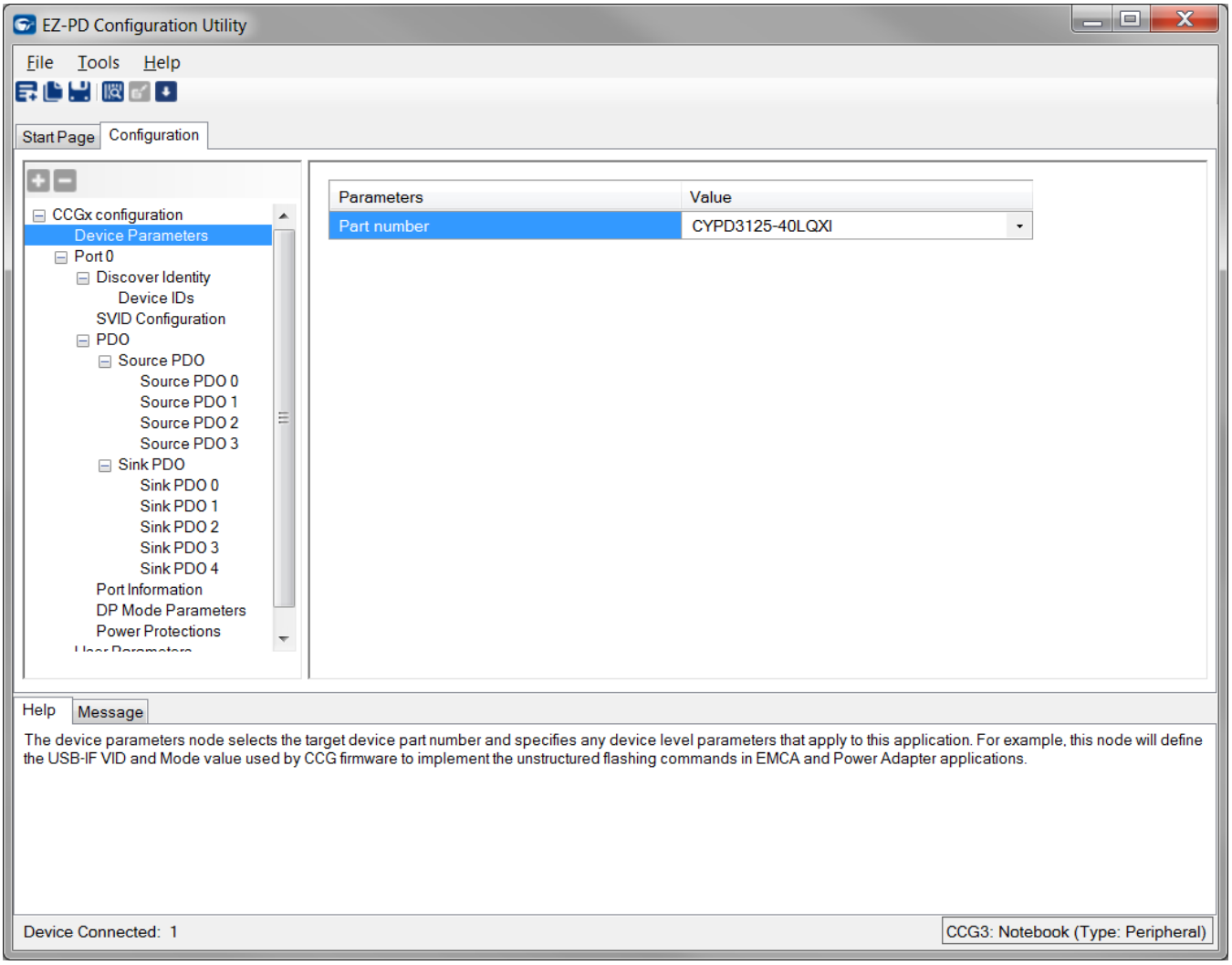
4. Select **File** > **Read from Device,** select **NOTEBOOK** in the USB-PD Device list as shown in Figure 19. Select the **Bootloader Read** checkbox and click on the **Read** button in the popup window to read the existing configuration as shown in Figure 19. The Bootloader Read checkbox causes the configuration to be read from the running firmware rather than the alternate firmware. Note that in your case, the running firmware may be FW1 and the alternate may be FW2. The version that is running does not matter in this case – the CCG device automatically runs whichever one was updated most recently.

Figure 19. Reading the Existing Configuration Table of the CCG3 Device of CY4531 CCG3 EVK



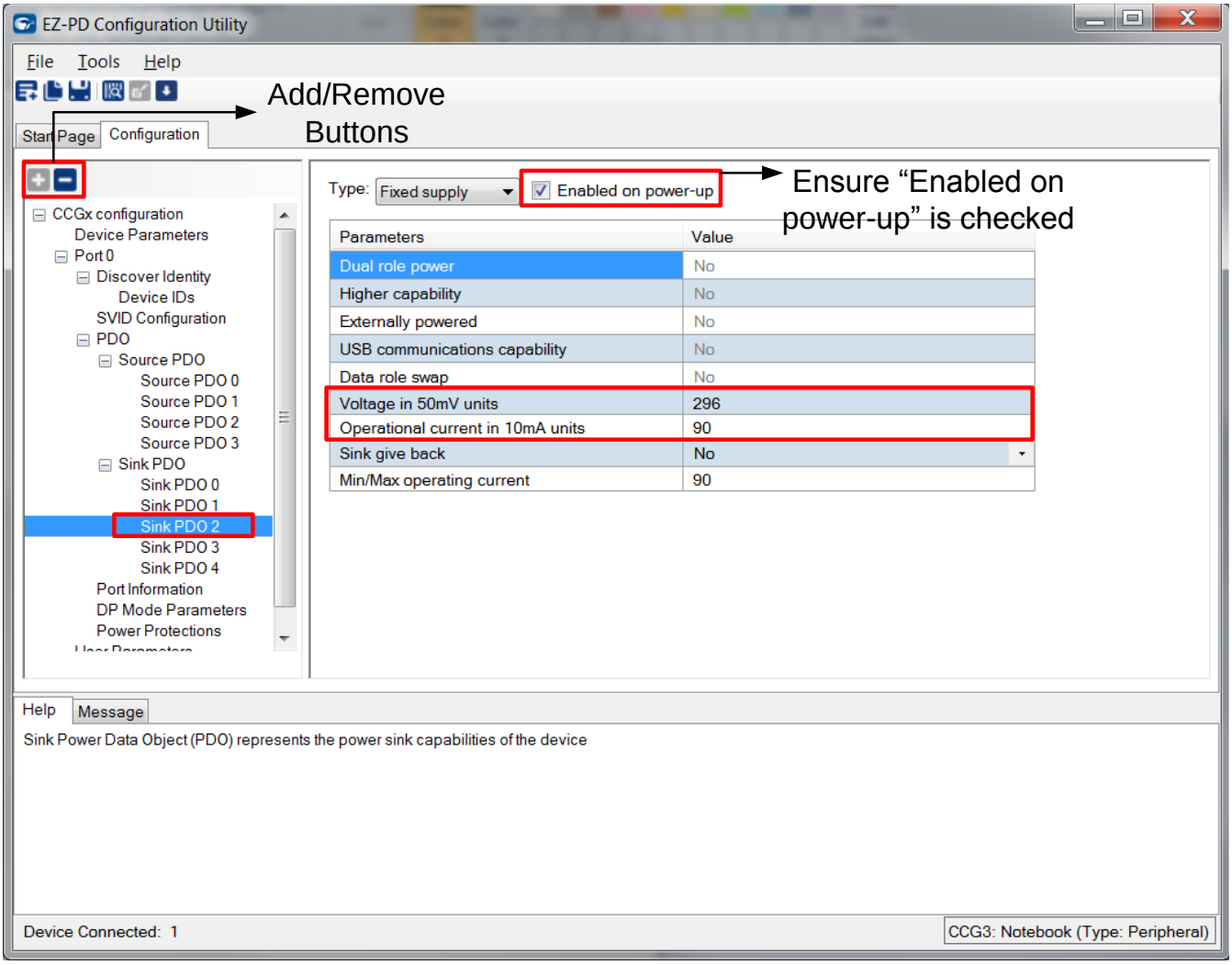
5. The existing configuration will now be read, and will be displayed in the utility in a tree under **CCGx Configuration** in the left panel as shown in Figure 20. Device configuration parameters are classified into Device Parameters, Port Parameters, and User Parameters. For multi-port devices, there will be multiple copies of the port parameter entries. Port parameters are further classified and grouped into a hierarchical tree structure. In this example, select **Device Parameters** in the left panel.

Figure 20. Existing Configuration Table of the CCG3 Device in CY4531 CCG3 EVK



6. Select the **Sink PDO** option in the CCGx configuration list for Port 0 (this corresponds to the Type-C port of the CCG3 EVK). It can be seen that the CCG3 supports five sink PDOs listed as Sink PDO 0 – 4 as shown in Figure 21 and the window on the right will show the values of voltage and current corresponding to the selected PDO. For example, clicking on Sink PDO 2 will show the corresponding power profile of 14.8V, 900mA (expressed in units of 50mV, 10mA) in the main window. That is, 296 * 50mV = 14.8V and 90 * 10mA = 900mA.

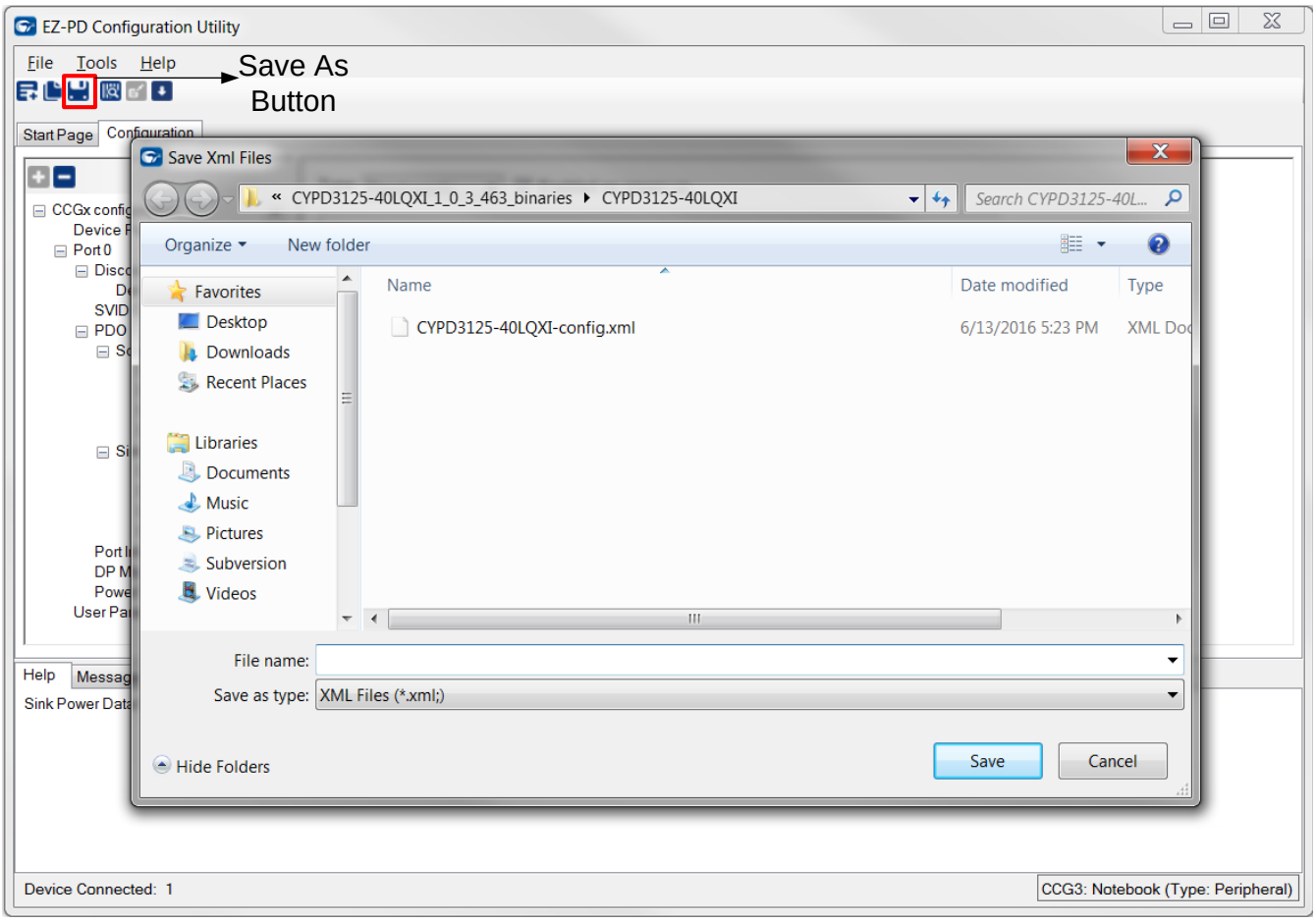
Figure 21. Sink PDOs Supported by CCG3 Device of CY4531 CCG3 EVK



7. Select the Sink PDO2 in the left panel and click on the **Remove (-)** button to delete it**.**
8. Click on the **Save As** icon to save the modified configuration as shown in Figure 22. The modified configuration is saved as an XML file. A *.cyacd* and a *.c* file will be saved in the same location.

For more details on the functionality related to the EZ-PD Configuration Utility, refer to its user manual by clicking here. You can also open the user manual by choosing **Help** > **User Manual** in the EZ-PD Configuration Utility, as shown in Figure 11.

Figure 22. Saving Modified Configuration in EZ-PD Configuration Utility



5.3 Program CCG3 Device with Updated Configuration Parameters Using the EZ-PD Configuration Utility

This section describes how to update the configuration parameters of a CCG3 device using the EZ-PD Configuration Utility. After the modified configuration table is saved, follow the steps below to update the configuration table in the CCG3 device using the EZ-PD Configuration Utility.

1. Ensure steps 1 to 8 of the Modify Configuration Parameters Using EZ-PD Configuration Utility section are successfully completed.
2. Select **Tools** > **Configure Device** as shown in Figure 23. A new window **Configure Device** opens where the target device to be programmed can be selected. Select **Notebook** as the target device, select the appropriate *cyacd* file to be programmed, and then click on the **Program** button as shown in Figure 24. Note that since the **Normal Flashing** option was selected as shown in Figure 24, the selected configuration file updates the configuration parameters of the firmware that is not running, i.e., the alternate firmware. This option also causes the running firmware to be switched at the end of the configuration update. Therefore, in this case, the alternate mode firmware will become the running firmware after the configuration update is complete and the device is reset.

Figure 23. Selecting “Configure Device” in EZ-PD Configuration Utility

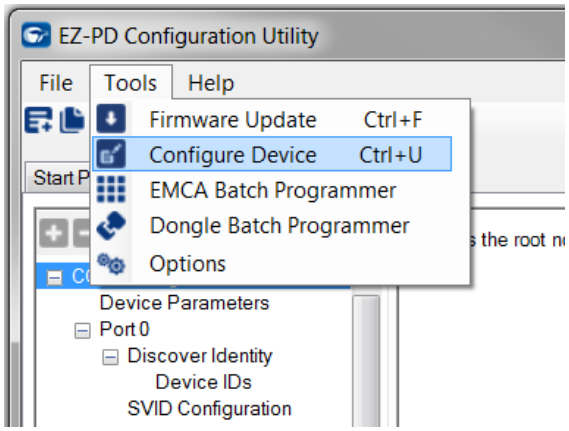
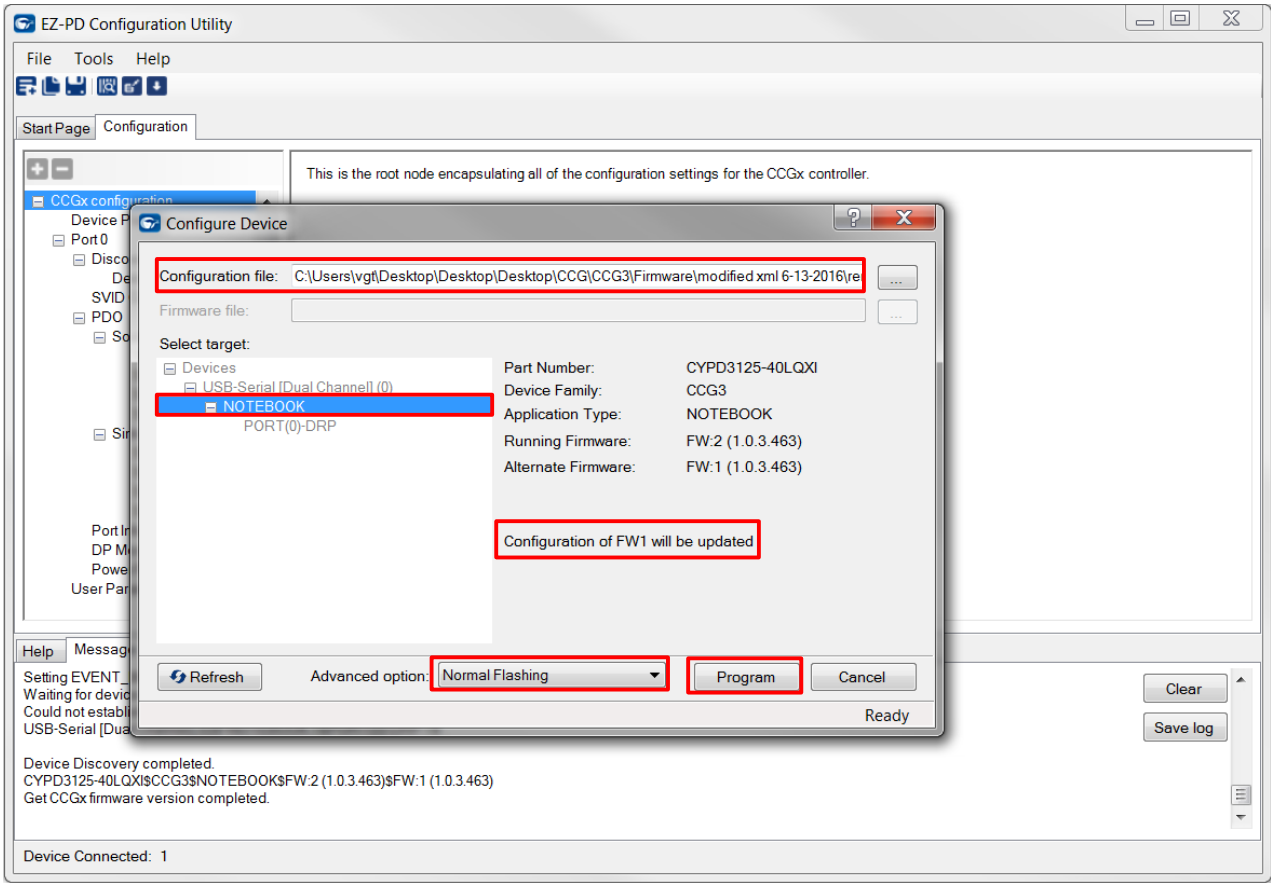
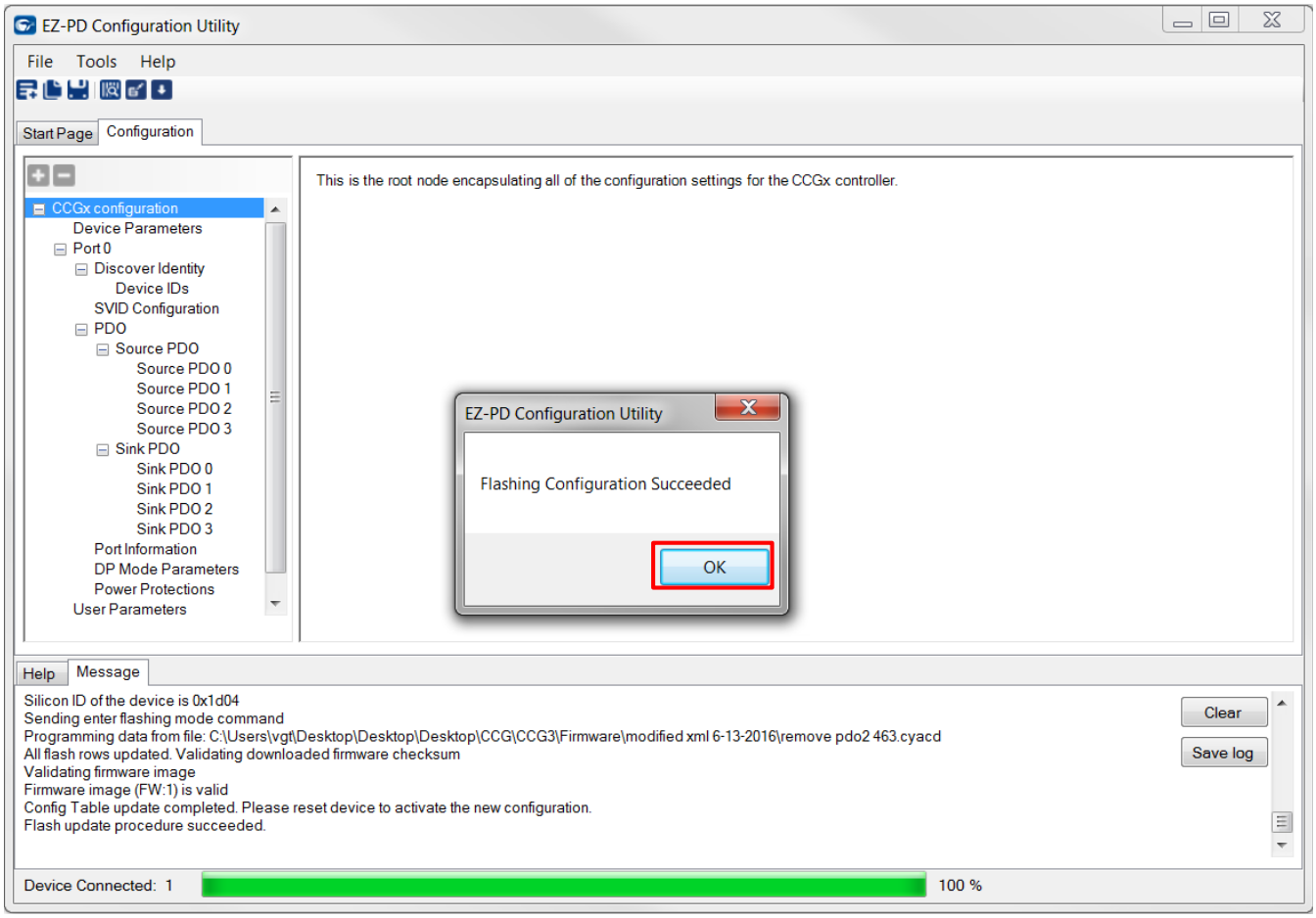


Figure 24. Updating CCG3 Device’s Configuration Table using EZ-PD Configuration Utility



3. The successful programming of the new configuration file will be indicated by a message box with the message **Flashing Configuration Succeeded**. Click **OK**. The **Message** window at the bottom of the utility also shows progress during the update process as shown in Figure 25. This shows that the configuration table of the CCG3 device of the CY4531 CCG3 EVK is successfully updated.

Figure 25. Message Showing Configuration Parameters Successfully Updated



4. Reset the CY4531 CCG3 EVK. This can be done by cycling the power to the kit or by pressing the reset switch SW1 on the CCG3 daughter card in order to reflect the modified configuration on the CCG3 device.
5. As mentioned previously, in the CCG3 application firmware, two copies of the firmware (FW1 and FW2) are used for fail-safe operation. One is called the running firmware and the other is called the alternate firmware. As shown in Figure 19, FW2 is the one that will be read because the **Bootloader Read** option is checked, which reads the running firmware. As shown in Figure 24, since **Normal Flashing** option is selected, the EZ-PD Configuration Utility updates the alternate firmware while the running firmware is being executed. Hence, it updates FW1 in this example. After the flash configuration is updated successfully and the device is reset, the CCG3 device bootloader is designed in such a way that the last updated firmware image becomes the running firmware. Hence, when the configuration table is read back as shown in Figure 26 with the **Bootloader Read** option selected, the FW1 image will be read, which is the firmware copy that was updated.
6. Repeat steps 4 and 5 of the Modify Configuration Parameters Using EZ-PD Configuration Utility section in order to read back the updated configuration table from the CCG3 device and verify. Notice that as shown in Figure 19, FW2 was read, as shown in Figure 24, FW1 was updated, and after recycling power from step 4 above, FW1 will be read as shown in Figure 26.
7. As shown in Figure 27, it can be seen that the CCG3 device of the CY4531 CCG3 EVK now only supports four sink PDOs listed as PDO 0 – 3, and not five PDOs as shown in Figure 20 earlier. Also note that PDO 2 no longer supports the power profile of 14.8V, 900mA.

Figure 26. Reading back Updated Configuration Table of CCG3 Device using EZ-PD Configuration Utility

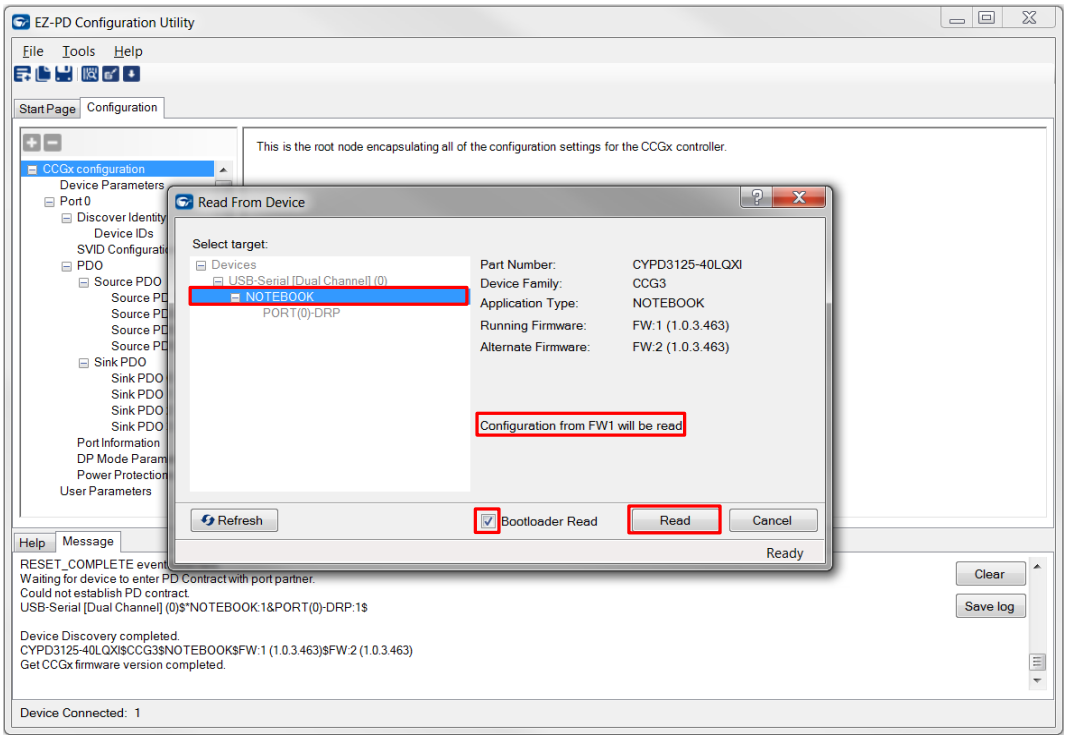
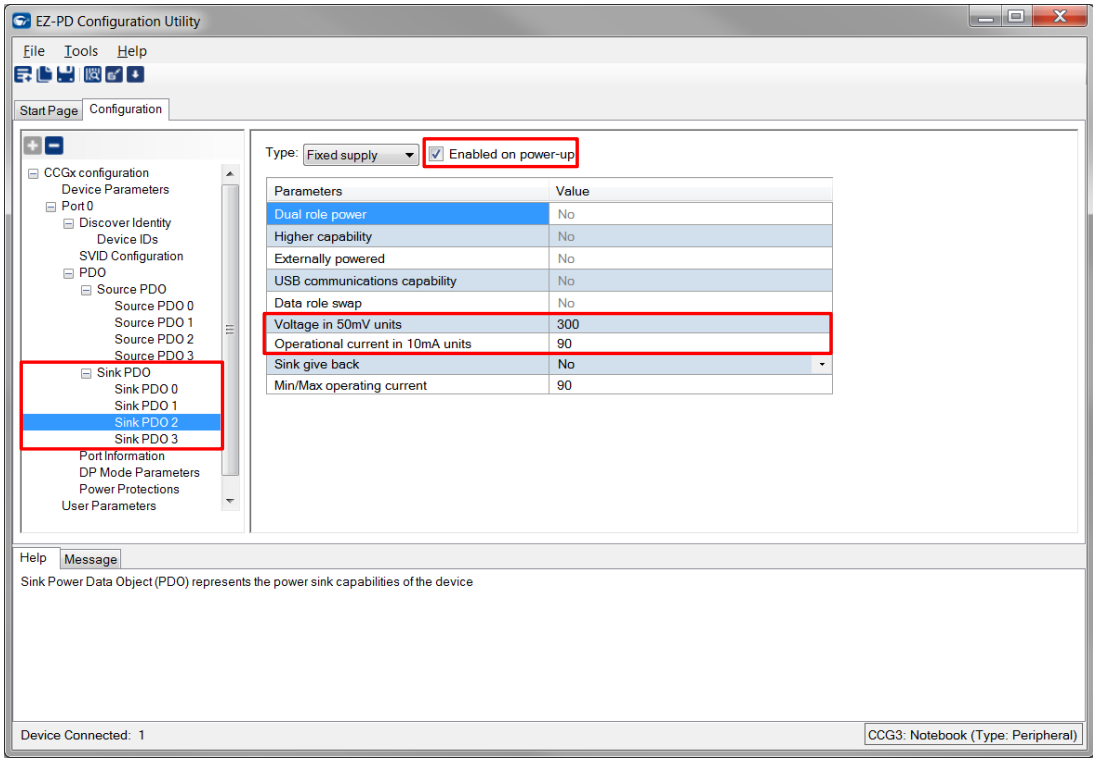


Figure 27. Verifying the Updated Configuration Table in CCG3 device using EZ-PD Configuration Utility

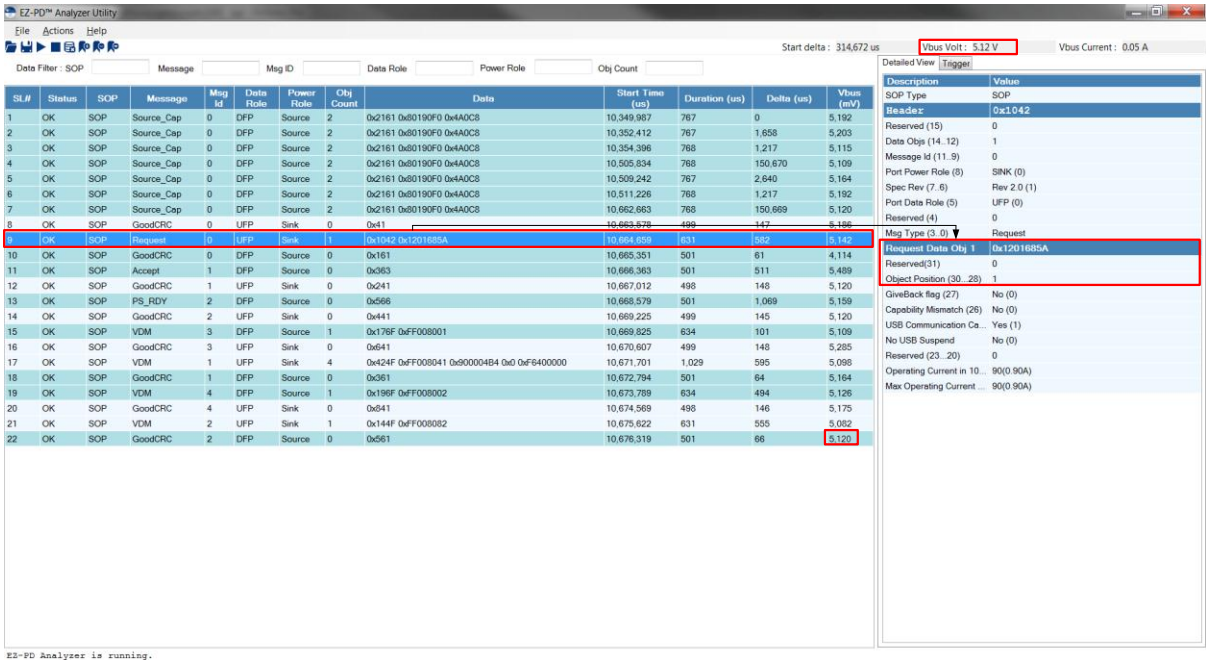


5.4 Re-test CY4531 CCG3 EVK Setup with Modified Configuration and Type-C Power Adapter

This section describes the steps to observe the behavior when the CY4531 CCG3 EVK with modified configuration parameters is connected to the Type-C power adapter. The same Apple 29W power adapter referenced in the earlier section is used in this section as well.

1. Disconnect all connections shown in Figure 17, and connect CY4500 EZ-PD Protocol Analyzer to the PC (USB Host) using a micro-USB cable. Ensure that jumper J4 of the CY4531 CCG3 EVK's daughter card is set to 2-3 position and its jumper J3 is set to 1-2 position.
2. Connect the Type-C plug of the CY4500 EZ-PD Protocol Analyzer to the Type-C port of the CCG3 EVK.
3. Launch the EZ-PD Analyzer Utility from **Windows** > **All Programs** > **Cypress** > **EZ-PD Analyzer Utility** > **EZ-PD Analyzer Utility** and click on the "**Start Capturing**" icon shown in Figure 14 to start capturing the CC traffic.
4. Connect the Apple 29W USB-C power adapter to the Type-C receptacle of the CY4500 EZ-PD Protocol Analyzer (as shown in Figure 15). Observe that the PD traffic is being captured on the EZ-PD Analyzer Utility.
5. From analyzing the CC traffic and looking at the V_{BUS} voltage at the end of the captured data as shown in Figure 28, it can be seen that a successful PD contract is established at 5V this time with the Type-C power adapter providing 5V to the Type-C port of the CCG3 EVK.

Figure 28. CC Trace Showing Power Contract Establishment @ 5V with Modified Configuration Parameters



6. Also, as seen from Figure 28, the CCG3 device requests the source PDO object 1, i.e., source PDO1. This implies that the custom PDO at 14.8V, 900mA (supported by the Type-C power adapter) is no longer supported as a sink PDO by the Type-C port of CY4531 CCG3 EVK, because the sink PDOs of the CCG3 device have been successfully updated.
7. This can also be verified by measuring the voltage on the power output header J7 of the CCG base board using a multimeter. The output voltage on this header will be about 5V, which is the established power contract between the two devices. Also, looking at the 'Vbus' column in Figure 28, it can be seen that the V_{BUS} voltage is set to 5V at the end of the power contract negotiation.

This shows that the power contract is successfully established at 5V between the Type-C power adapter and the CY4531 CCG3 EVK, and not at 14.8V. The Type-C power adapter provides 5V to the Type-C port of the CY4531 CCG3 EVK. In order to restore the CY4531 CCG3 EVK back to the default configuration parameters, follow the steps described in section “Programming the CCG3 Device on CY4531 CCG3 EVK” of the CY4531 EZ-PD CCG3 EVK Guide. Once completed, the steps described in the Test CY4531 CCG3 EVK Setup with the Default Configuration and Type-C Power Adapter section can be repeated to verify the kit operation with default configuration parameters.

Document History

Document Title: AN200210 – Getting Started with EZ-PD™ CCG3

Document Number: 002-00210

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	5320774	VGT	06/23/2016	New application note.
*A	5857634	HARA	08/18/2017	Updated logo and copyright.

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6

Cypress Developer Community

Forums | WICED IOT Forums | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

PSoC is a registered trademark and PSoC Creator is a trademark of Cypress Semiconductor Corporation.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709