UM2546
User manual

STM32G071B-DISCO USB-C Discovery kit

Introduction

The STM32G071B-DISCO USB-C Discovery kit comes on top of the STM32Cube as a firmware package that offers a full set of software components based on a modular architecture allowing each module to be used separately in standalone sink applications.

The STM32G071B-DISCO USB-C Discovery kit may be executed in two different modes depending on the position of the switch: standalone sink mode, or spy mode. The spy mode does not use the USBPD stack. In this mode, the protocol information on the CC lines is decoded, and no protocol action may be triggered. But in the standalone sink mode, some protocol actions may be executed.

In both modes, the UCPD block port 1 is used, and may give an example for a customer application.

The STM32G071B-DISCO USB-C Discovery kit has been designed to run on STM32G071B-DISCO (MB1378).

Figure 1. STM32G071B-DISCO USB-C Discovery kit

Contents

List of tables

List of figures

1 Demonstration description

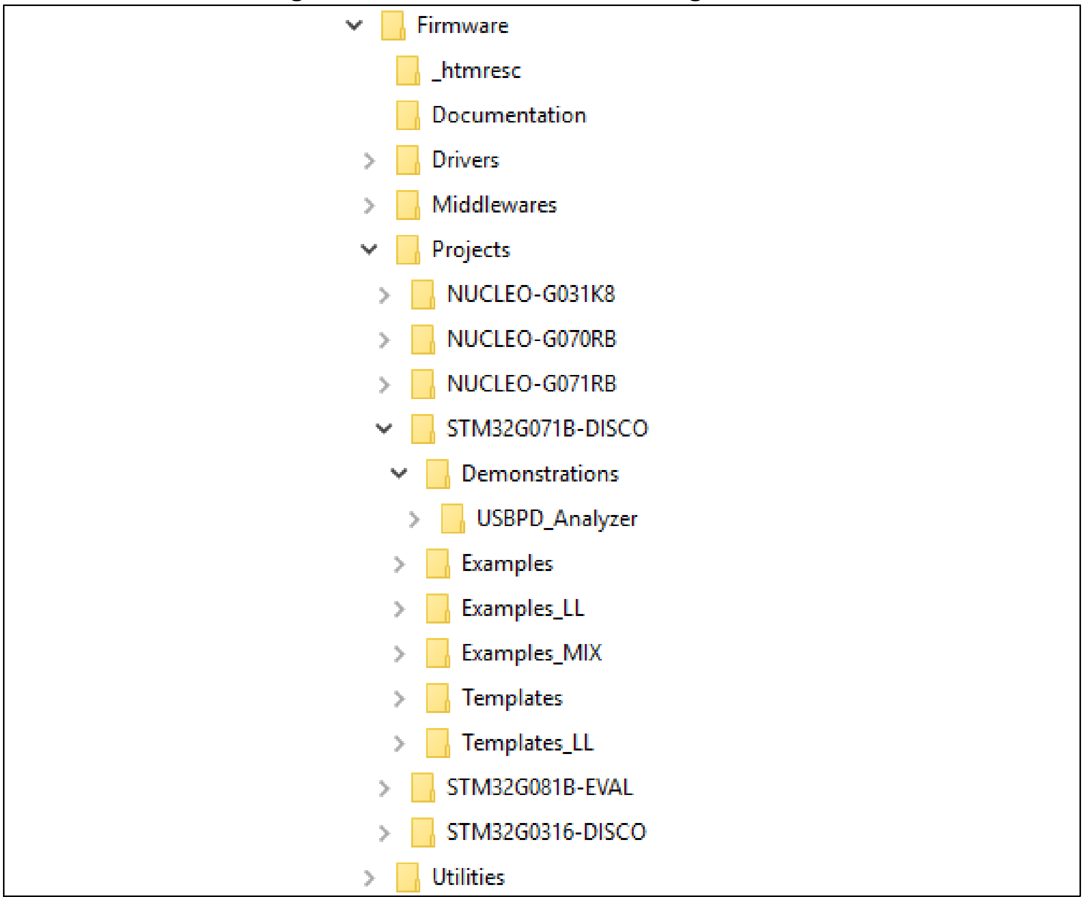
The demonstration has been designed with the following objectives:

- Toolkit with low memory consumption
- Modular applications: independents with high level of reuse
- Basic menu navigation through joystick
- Comprehensive G0 functional coverage

1.1 Demonstration package

Figure 2 shows the demonstration folder organization:

Figure 2. Demonstration folder organization

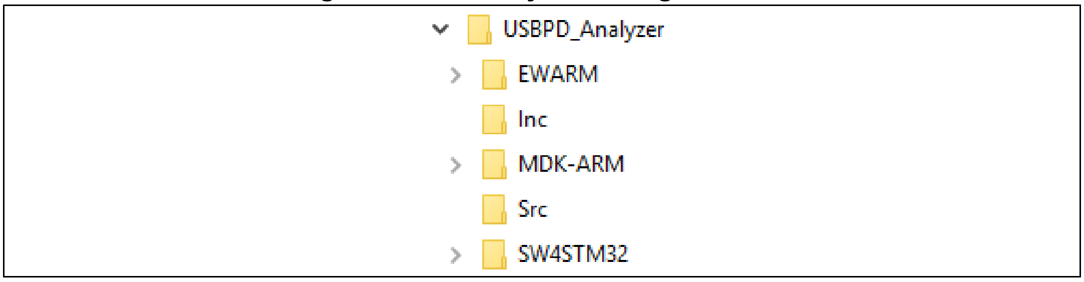


The demonstration sources are located in the "Projects" folder of the STM32Cube package for each supported board, here in STM32G071B-DISCO folder.

The demonstration firmware aims at demonstrating how USB-PD version PD3.0 has been implemented in the context of STM32G0xx devices.

Figure 3 illustrates the organization of the UCPD folder.

Figure 3. Discovery folder organization



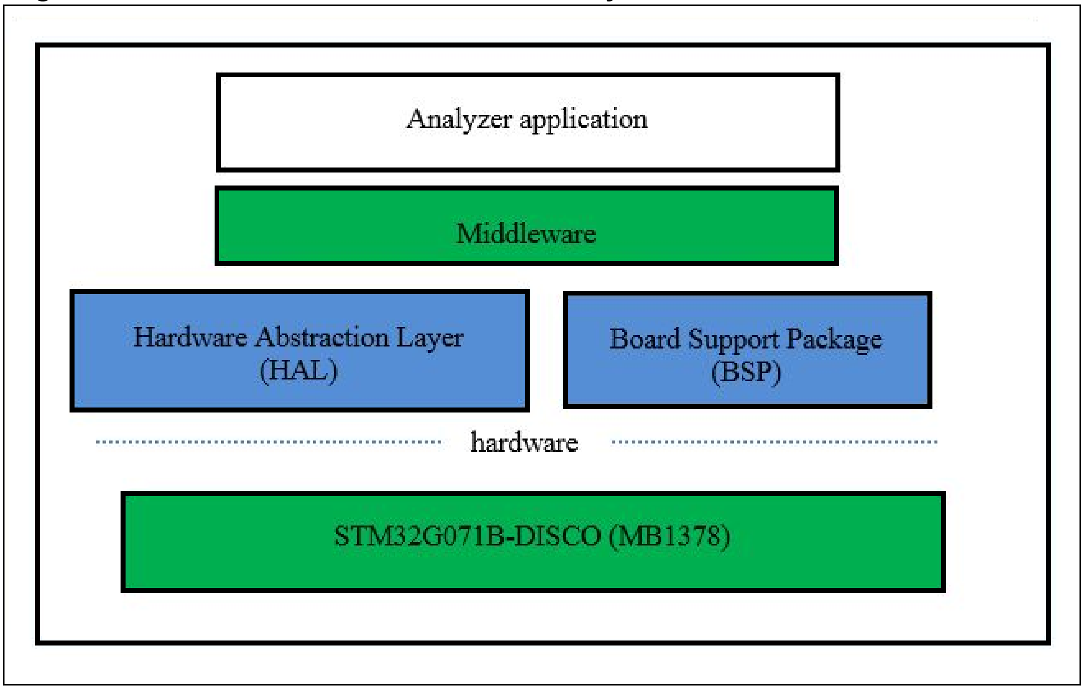
UCPD_Analyzer sub-folders:

- Inc: Analyzer application header files
- Src: Analyzer application implementation
- Software development environments:
  - EWARM: IAR embedded workbench
  - MDK ARM: Keil Microcontroller Development Kit
  - SW4STM32: System workbench for STM32

1.2 Demonstration architecture overview

The top level software architecture of the STM32G071B-DISCO USB-C Discovery kit firmware is represented on Figure 4. The software elements mentioned in this diagram are briefly depicted in dedicated sections.

Figure 4. STM32G071B-DISCO USB-C Discovery kit firmware - Software architecture



1.2.1 Analyzer

The UCPD Analyzer application uses only one G0 port which mainly consists in Type-C connection/disconnection detection and Type-C power contract negotiation.

1.2.2 HAL level

HAL level layer consists in the stm32g0xx.HAL drivers together with the STM32G071B-DISCO board support package (BSP).

1.2.3 Middleware

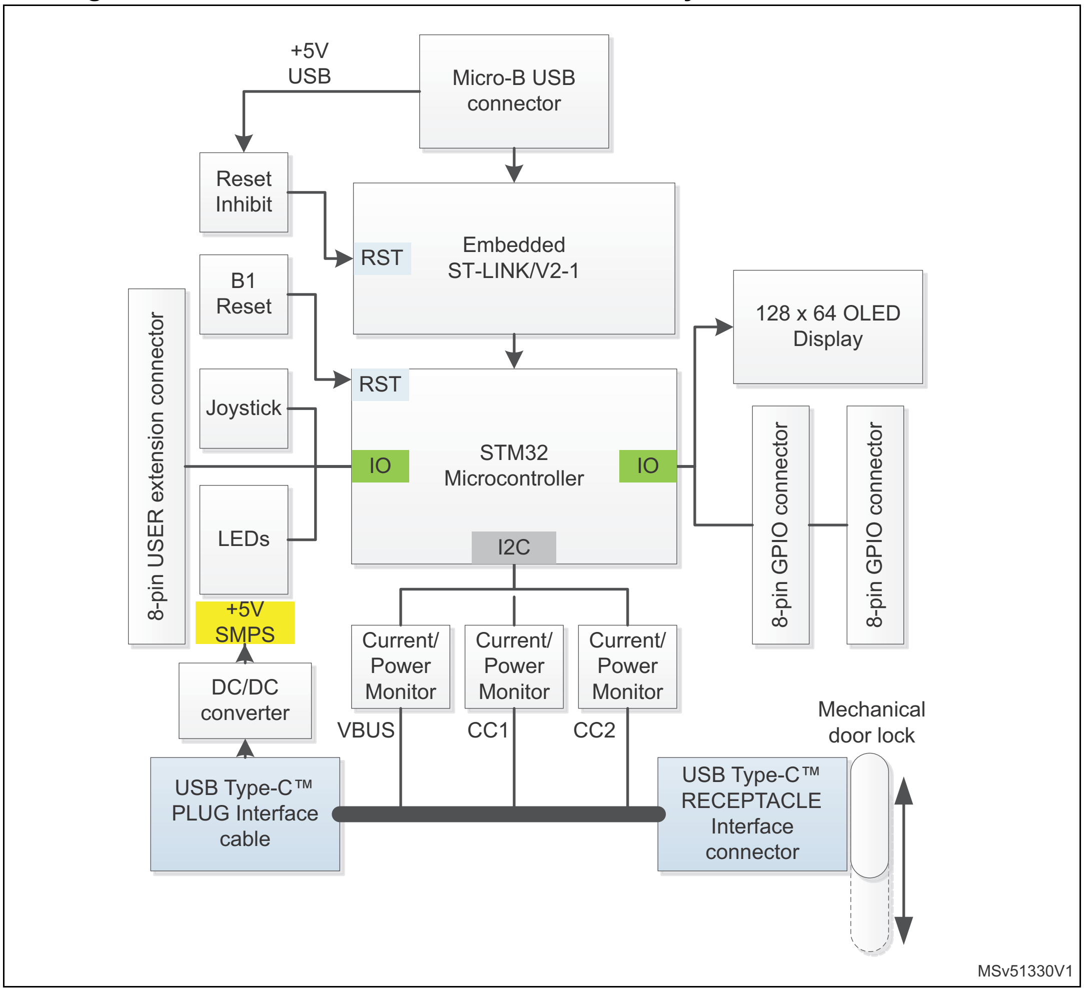
The middleware provides the following modules:

- FreeRTOS: FreeRTOS open source solution. UCPD application is based on FreeRTOS.
- USBPD: USB-PD software stack

1.3 STM32G071RB resources

1.3.1 Hardware resources used by the UCPD demonstration

Figure 5. STM32G071RBT hardware blocs used by the UCPD demonstration



1.3.2 Peripherals used by the UCPD demonstration

Figure 6. STM32G071RBT peripherals used by the UCPD demonstration

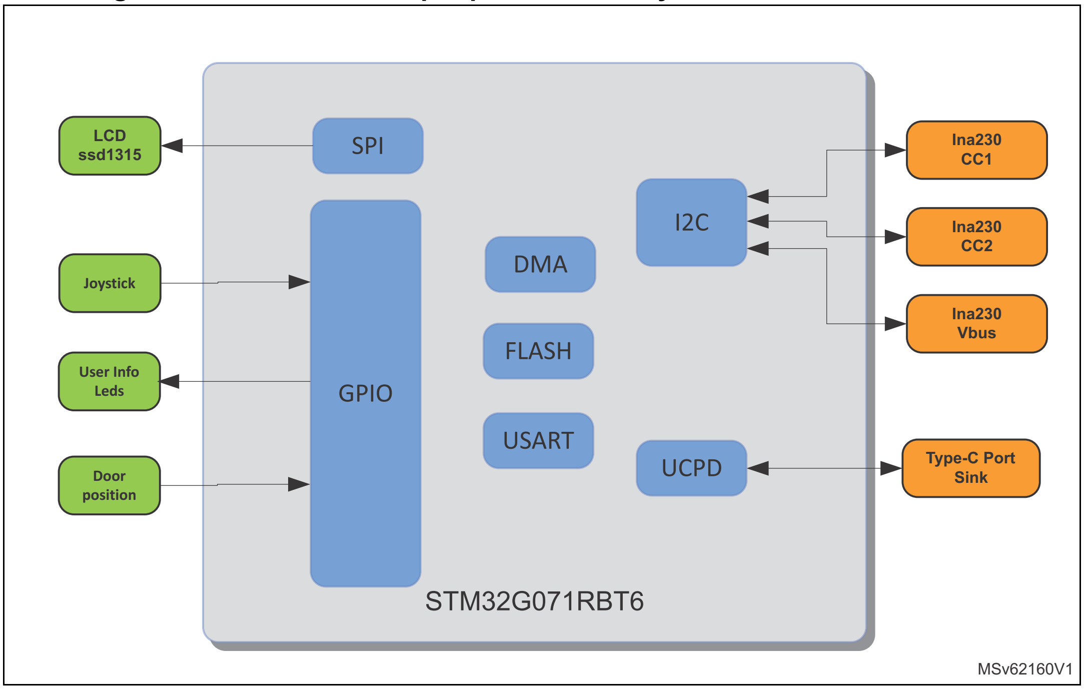


Table 1. STM32G071RBT peripherals used by the UCPD demonstration

Peripheral	Usage description
SPI	LCD is controlled through SPI1. Write accesses to the LCD are performed to display strings and bitmaps during the UCPD demonstration execution.
GPIO	The GPIO pins connected to the joystick are used to interact with the UCPD demonstration (e.g. menu navigation). One GPIO pin is used to detect the door position Several GPIOs are used to drive some indication LEDs
I2C	I2C1 is used to control 3 different INA230: – Two INA230 to detect the voltage level on both CC lines – One INA230 to monitor the voltage and the current on VBUS
UCPD	UCPD1 is used to manage the USB Type-C communication over the Type-C port.
DMA	DMA is used for ADC conversions.

1.3.3 Interrupts

Table 2 shows all the external interrupts used by the demonstration

Table 2. STM32G071RBT demonstration interrupts usage

Interrupt	Usage description
Systick	Delay management
EXTI line 0	Joystick SEL (interrupt mode, rising edge)

Table 2. STM32G071RBT demonstration interrupts usage (continued)

Interrupt	Usage description
EXTI line 2	Joystick UP (interrupt mode, rising edge)
EXTI line 3	Joystick DOWN (interrupt mode, rising edge)
EXTI line 7	Joystick RIGHT (interrupt mode, rising edge)
EXTI line 1	Joystick LEFT (interrupt mode, rising edge)
EXTI line 13	Tamper (interrupt mode, rising edge)
DMA1 Channel1	DAC/ADC conversions completion
ADC1_COMP	ADC analog watchdogs
UCPD	UCPD related interrupts (e.g. Rx message received, Rx ordered set detected, Transmit message sent, …)

2 Demonstration functional description

2.1 Demo startup

2.1.1 Normal processing

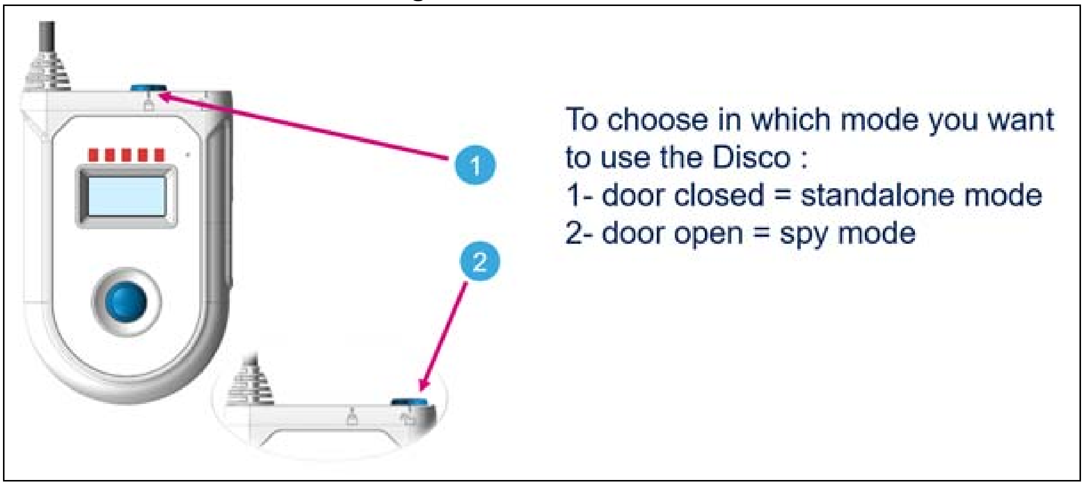
If the STM32G071B-DISCO is powered from the micro USB, after a board reset, at demo startup the welcome screen is displayed.

2.2 UCPD demonstration

2.2.1 Mode selection

Depending on the position of the switch, the analyzer may run in two modes:

Figure 7. Mode selection



2.2.2 LED indications

The mode selection is confirmed by the LED LD6:

Figure 8. LED indications

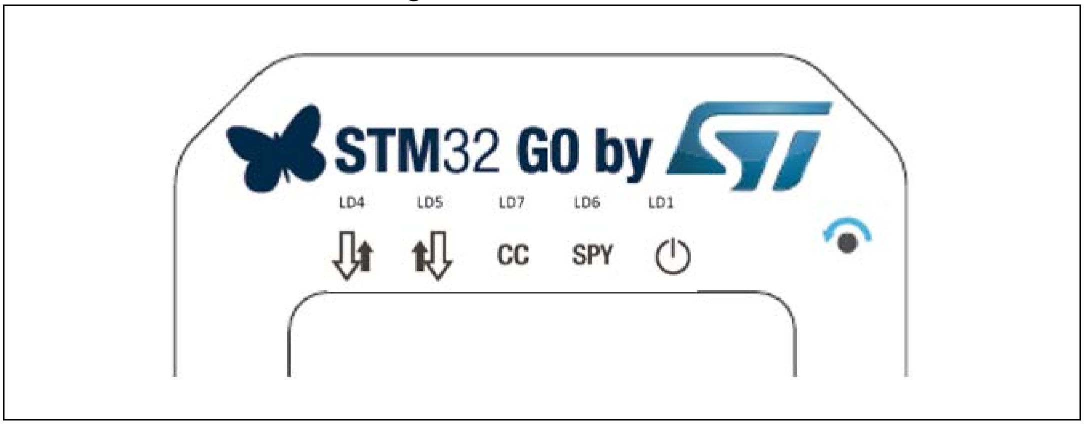


Table 3. LED assignation

Reference	Color	Name	Function
LD1	Green	POWER 5V	5V present onto the board
LD4	Orange	SINK mode	Attached to a power source
LD5	Orange	SOURCE mode	Attached to a device as source
LD6	Green	SPY mode	SPY mode active
LD7	Green	CC	CC1 active line

3 Hardware settings

On this kit, no particular setting must be configured.

The G0 Discovery kit may be powered on with a micro USB cable, or it may power itself using the VBUS, when connected to a type C Source.

For detailed information on the hardware part, please check *STM32G0 Discovery kit for USB Type-C™ and Power Delivery* user manual (UM2401)

3.1 STM32G071B-DISCO board

The STM32Cube demonstration supports STM32G071RB device and runs on STM32G071B-DISCO board from STMicroelectronics.

Figure 9. STM32G071B-DISCO top view

Figure 10. STM32G071B-DISCO bottom view

4 Software settings

4.1 Clock Control

STM32G071 internal clocks are derived from the HSI running at 16Mhz.

In this demo application, the various system clocks are configured as follows:

- System clock is set to 64 MHz: the PLL is used as the system clock source.
- HCLK frequency is set to 64 MHz.

4.2 Programming the demonstration

The user may program the demonstration using two methods:

4.2.1 Using Binary file

To program demonstration's binary image into the internal Flash memory, the user may exercise STM32G071B-DISCO_USBPD_Analyzer.hex file, thanks to ST-Link Utility or STM32CubeProgrammer.

Please refer to the Binary Resources Demo in the Board web page STM32G071B-DISCO (https://www.st.com/en/evaluation-tools/stm32g071b-disco.html).

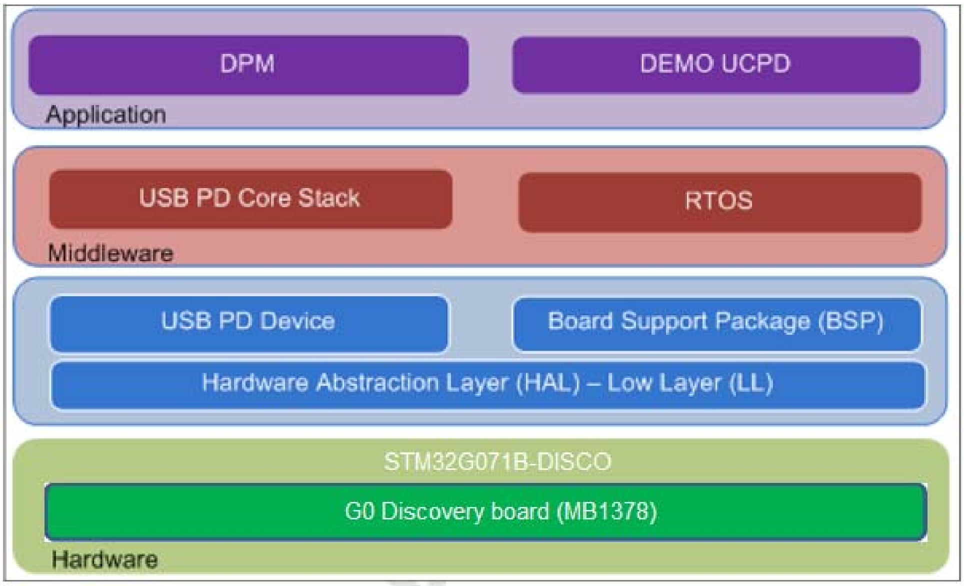
4.2.2 Using preconfigured projects

Select the folder corresponding to the preferred toolchain (MDK-ARM, EWARM or SW4STM32).

- Open USBPD_Analyzer project and rebuild all sources.
- Load the project image through the debugger.
- Restart the evaluation board (press B1: reset button).

5 Software description

Figure 11. STM32G0 software architecture overview



Following sections detail:

- Application: in charge to initialize demo application, HAL, interrupt handler and launch the main module
- Middleware: FreeRTOS, USBPD

5.1 Demonstration Application

The application goal is to prepare demonstration startup, by initializing all the HW/SW.

Table below provides a description of all the actions performed by the different functions in main.c

Table 4. Main application functions description

Functions (Main.c)	Description
Main	Initialize the HAL, configure the clock and the power MOS, depending on the door position, initialize the responder for the STM32CubeMonitor USB Type-C PD tool (GUI_Init), start the stack (USBPD_DPM_InitCore), the UCPD application (USBPD_DPM_UserInit), start the OS (USBPD_DPM_InitOS), and then start the demonstration
SystemClock_Config	Set the right clocks for flash and RCC

The file "stm32g0xx_it.c" is also part of the application and is used, as usual, to map the interrupt vector on the driver HAL driver, depending on the module requirement (for debug trace, and joystick management)

Main demonstration functionalities are in the file demo_disco.c.

Table 5. Main applications functions description

Functions (demo_disco.c)	Description
DEMO_Manage_spy	This is the main spy function. It gets the VBUS voltage and current from the INA230, and gets the voltage level of both CC lines. If the message queue has some messages to process, Display_add_spymsg is called
DEMO_Manage_event	Main function for standalone mode.
Display_add_spymsg	Function goal is to fill the USB PD data structures with the information received from the CC lines. These structures are then used by the application, to feed the G0 disco information display.
DEMO_SPY_Handler	This function is called on any UCPD event thanks to an interrupt. It posts the USBPD protocol message seen from the CC lines to a queue. Later, the application reads this queue and decode the protocol messages. We don't decode the message under interrupt state, to be ready as soon as possible to see the next messages on the CC lines.
DEMO_InitBSP	Initialise the LCD, and the voltage monitoring of VBUS and CC lines.
Check_cc_attachement	Used in case of spy mode: to detect which CC line is used for power delivery protocol exchanges.
DEMO_PostMMIMessage	Function that posts the joystick press event in a FreeRTOS queue, thanks to HAL_GPIO_EXTI_Rising_Callback function.
*_menu_nav	Function that is used to manage the joystick up and down press to navigate into a specific menu (source power profiles, sink power profiles, extended capabilities, command…)
*_menu_exec	Function that executes the selected specific action (select power profile, command…)
Intialize_RX_processing	Configures the UCPD IP (DMA, Interrupts) to be ready to capture UCPD messages.

5.2 Application overview

Depending on the door position (GPIO DOOR_SENSE_PIN: PC8), two different sub-applications may be executed.

In the spy case, USBPD stack is not launched. INA230 is used to know which CC line is selected for communication, and then decode the messages on it.

In the standalone mode, the full USBPD stack is running.

5.2.1 Spy mode case

In spy mode, the door is open, and the Discovery kit may receive a second plug to spy the messages exchanges.

In spy mode, the Discovery kit must not interfere with the two type C devices to which it is connected.

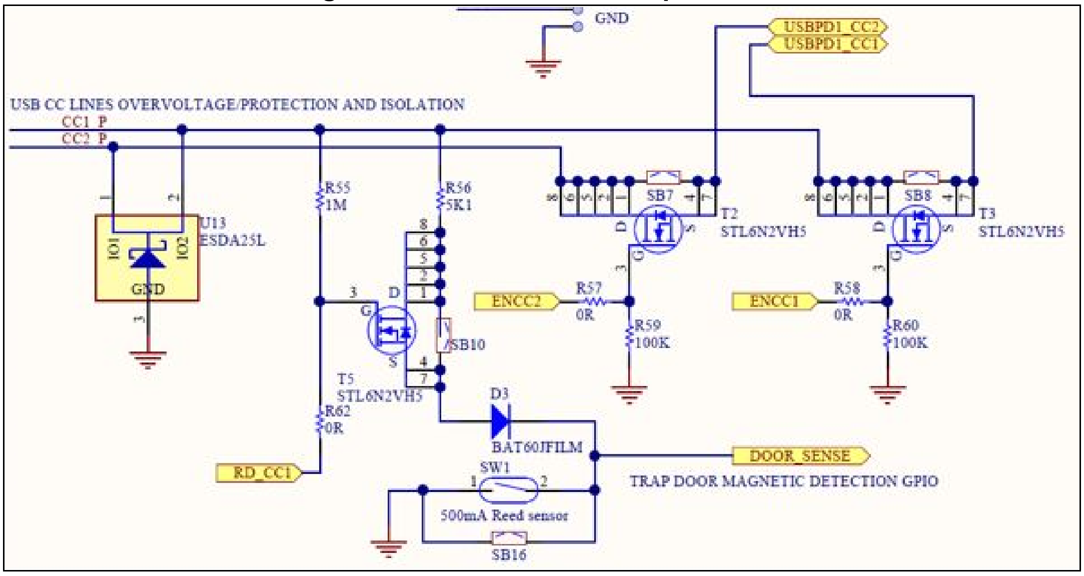
This is why, as soon as open door is detected, T2, T3 and T5 MOSFETs are turned ON.

Table 6. Code example: MOSFET control in spy mode

```
BSP_MOSFET_On(MOSFET_ENCC1);
BSP_MOSFET_On(MOSFET_ENCC2);
```

Door is open, RD56 (5K1 Ohms) is ignored (needed only for standalone mode to be seen as a sink), and CC1 and CC2 pins are connected from the G0 UCPD IP.

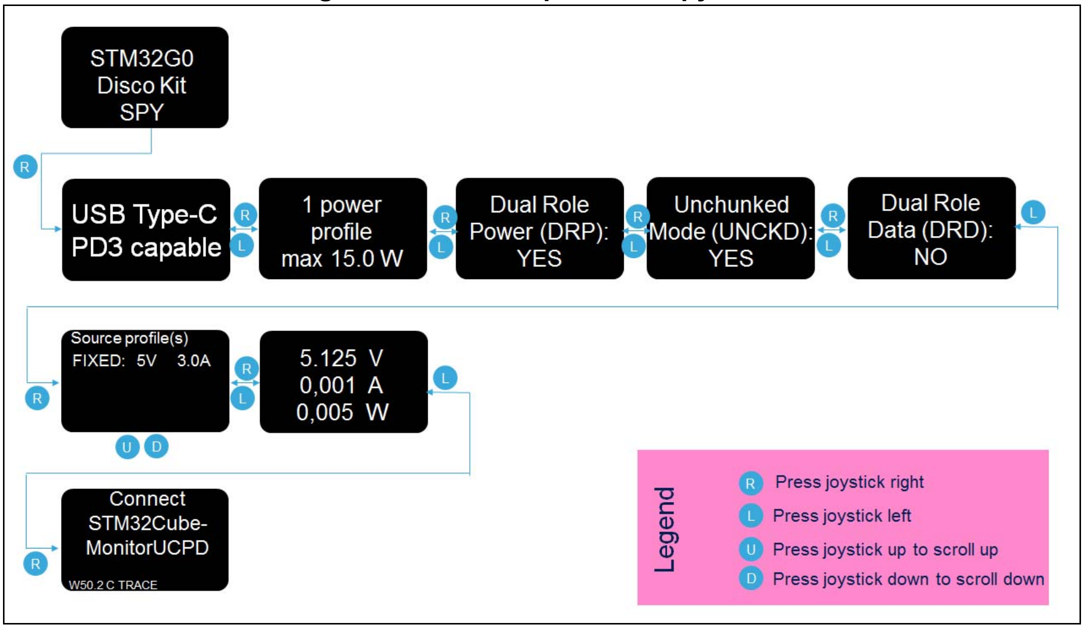
Figure 12. CC lines isolation/pull down



if no micro USB is plugged, as soon as the source device detects the sink device, VBUS is driven, and the G0 Discovery kit is powered on.

Here are the menus sequence. Details on the menus in following *Section 5.2.3*

Figure 13. Menu sequence in spy mode



5.2.2 Standalone sink case

In standalone mode, the board is in captive cable configuration. Only CC1 line is used for power delivery protocol exchanges.

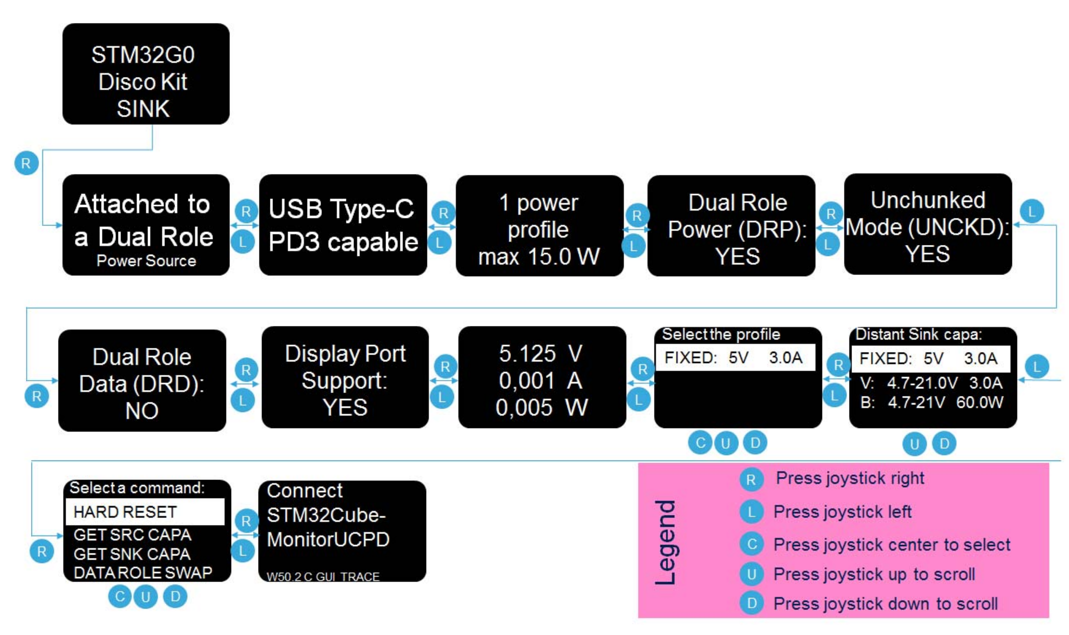
In this mode, application may boot from the VBUS provided by the source. So dead battery indication may be provided and Rd resistors on the CC1 line must be shown.

This is why when the door is closed, the T5 MOSFET is enabled and RD_CC1 signal is driven so that the resistor R56 is set between CC1_P and GND. See details in *Figure 12*.

In the standalone spy case, some specific actions may be requested using the joystick (select the power profile…), thus menus sequence is a little bit different from the one in spy mode configuration.

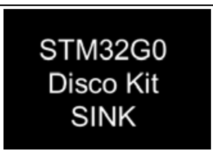
Details on the menus in following *Section 5.2.3*

Figure 14. Menu sequence in standalone sink mode



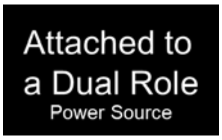
5.2.3 Demonstration menu details

Figure 15. Welcome screen



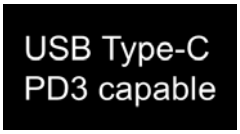
When the G0 disco is powered on (if micro USB is plugged or VBUS driven on the type C cable) this screen is visible.

Figure 16. Attach screen



Whenever a VBUS is present on the type C cable, this information is displayed. Then, depending on the source capabilities seen, dual role information may be displayed.

Figure 17. PD capability screen

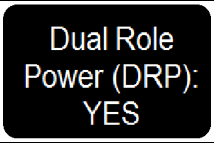


If USBPD protocol exchanges are seen over the CC lines, the power delivery specification version is displayed. This information may appear in spy mode, even if one device doesn't support power delivery

Figure 18. Power screen

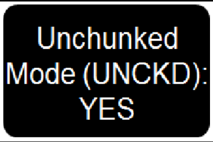
When the source capabilities are seen over the CC lines, the maximum power profile is displayed.

Figure 19. Power role screen



This screen will indicate if the power role swap is supported by the connected source device.

Figure 20. Unchunked mode support screen



This screen will indicate if the connected source device supports unchunked messages.

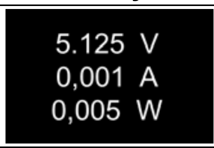
Figure 21. Data role screen

This screen will indicate if the connected source device supports data role swap.

Figure 22. Display Port screen (only standalone mode)

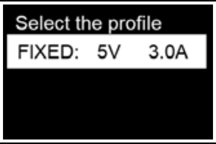
If the source is capable of using the super speed lines to send display port video signal, the information is displayed. This screen also displays if Thunderbolt capability is available. This information is gathered only in standalone sink mode, because the G0 issues a Discovery identity message. In some case, the source may not respond to this message, and therefore no information may be retrieved.

Figure 23. Power delivery information screen



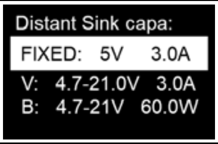
This screen displays the current VBUS power: voltage and current, and calculated power.

Figure 24. Source profile screen



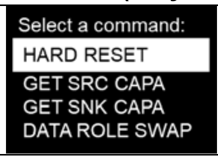
This screen displays all source power profiles. The user scrolls the list via the joystick. In case of standalone sink mode, he may request a dedicated power profile using the joystick (center click on the desired profile)

Figure 25. Distant Sink profile screen (only standalone mode)



This screen displays the sink profiles. This information is retrieved only in standalone sink mode, because a dedicated command (get sink capabilities) is sent by the G0 disco application. No actions or selection may be triggered here on a distant sink profile.

Figure 26. Action screen (only standalone mode)



On this screen, the user may execute the displayed commands using the joystick. (center to select). He may scroll up or down to access all possible actions.

Figure 27. Version screen

This screen displays the current G0 Discovery demonstration version.

We remind the user that the STM32CubeMonitor USB Type-C PD tool application may be run on a PC to get more details about the USBPD protocol messages that are exchanged. See chapter 5.2.4.

5.2.4 STM32CubeMonitor USB Type-C PD tool with G0 Discovery kit

We recommend to always plug the micro USB cable so that the disco is powered on, and have the possibility to see the details on the USBPD protocol exchanges.

In case of spy mode, as no action may be requested using the G0 disco, only the trace is available.

Figure 28. STM32CubeMonitor USB Type-C PD tool debug trace selection

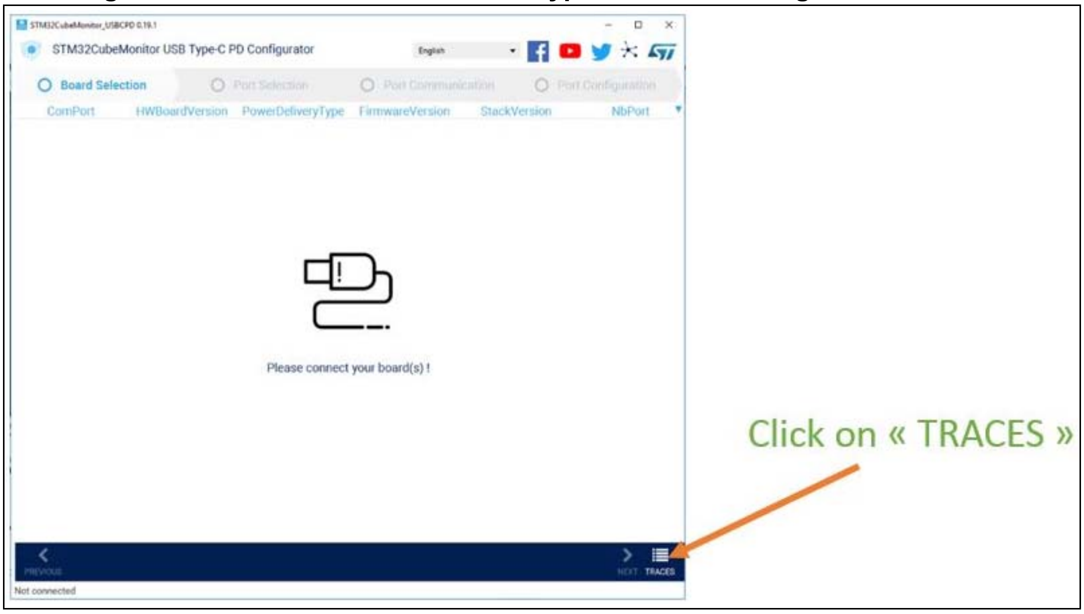
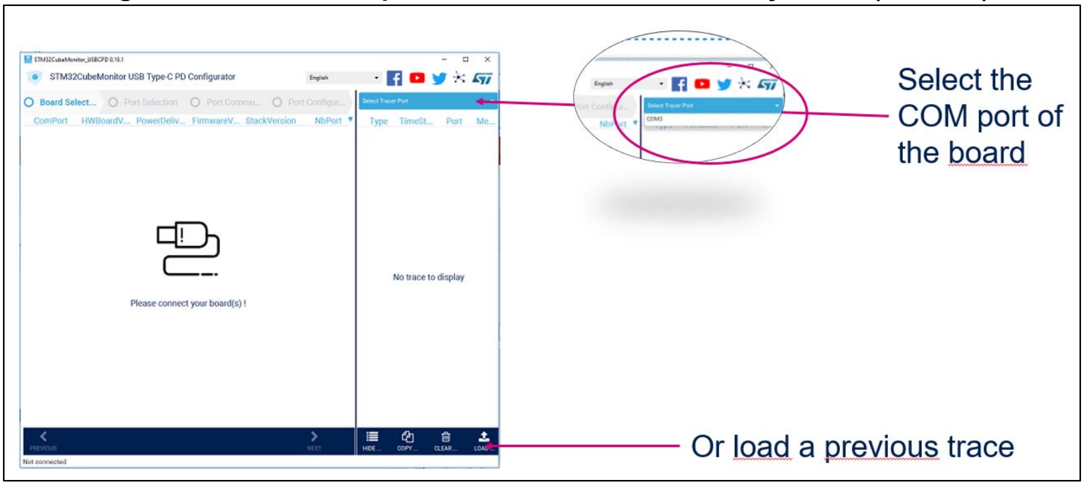
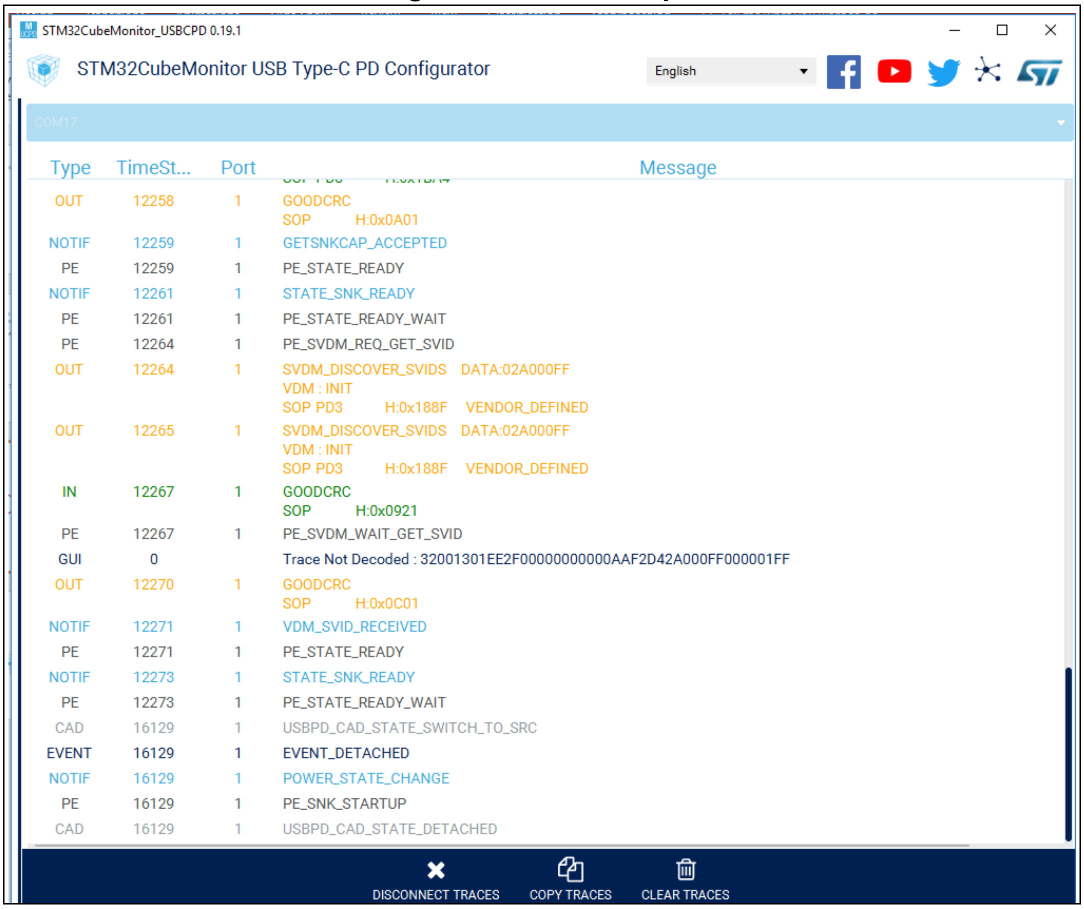


Figure 29. STM32G0 top and bottom view of Discovery board (MB1378)



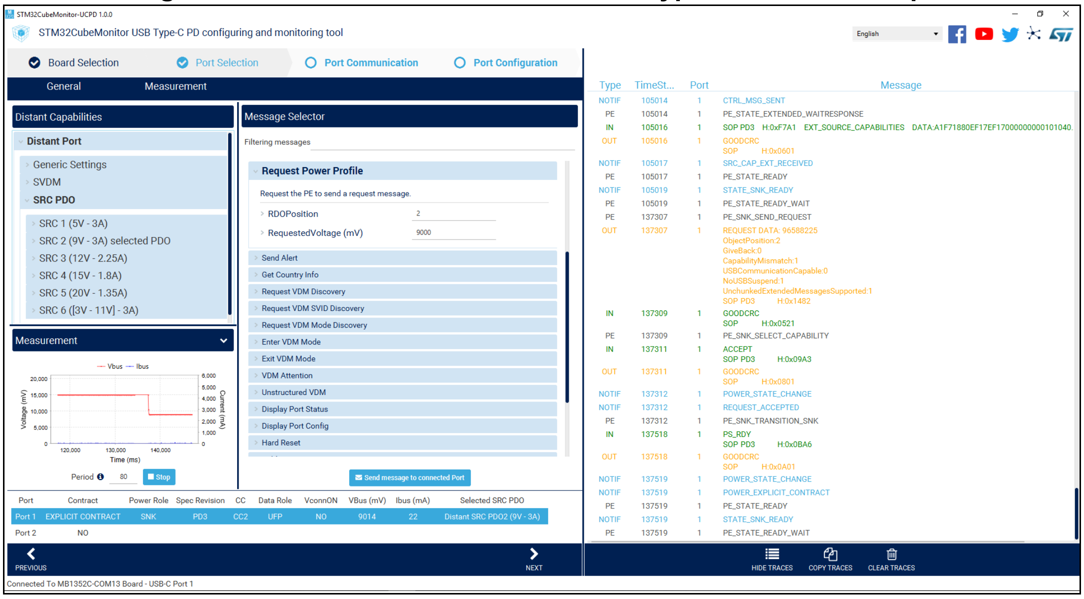
The user gets such a screen:

Figure 30. Trace example



When the G0 disco application is used in standalone sink mode, the full STM32CubeMonitor USB Type-C PD tool application (GUI) may be used to issue some commands like a data role swap for example.

Figure 31. Full STM32CubeMonitor USB Type-C PD tool example



For more details see *STM32CubeMonitor-UCPD software tool for USB Type-C™ Power Delivery port management* user manual (UM2468).

5.2.5 Dynamic memory use

The demonstration is currently using:

- CSTACK = 0x300
- HEAP = 0x800

5.3 Middlewares

5.3.1 USB PD

In this application the USBPD stack used is delivered as a library.

All information regarding this library may be found in the *STM32Cube USBPD stack* user manual (UM2552).

5.3.2 FreeRTOS

FreeRTOS: FreeRTOS open source solution.

More details may be found at: https://freertos.org

6 Footprint

This chapter sums up Ram / Rom consumption per software blocks.

Here is a full demonstration of the software consumption:

Table 7. RAM/ROM consumption

Full demonstration software	Read only code memory [Byte]	Read only data [Byte]	Read/write data memory [Byte]
STM32CubeMonitor USB Type-C PD tool responder (GUI)	7682	-	540
Demo Application	13596	2345	842
Policy Engine (PE)	**20320**	**16**	**20**
Stack Protocol layer (PRL)	**3000**	**2**	**12**
Cable detection (CAD)	**1484**	**-**	**24**
Drivers	2254	8802	324
HAL	4706	55	40
Physical	1360	-	72
Debug- Trace	280	-	-
FreeRTOS	5000	1	15628
Standard Libraries	5396	78	5029
G0 Device	1948	22	65
Grand Total	67026	11321	22596

7 Acronyms

Table 8. Table of acronyms

Acronym	-
CC	Configuration channel
USB PD	USB Power Delivery
GUI	Graphical User Interface: UCPD PC monitor application
PD	Power Delivery
DRP	Dual Role Power
FRS	Fast Role Swap
DRD	Dual Role Data
VDM	Vendor Defined Message

8 Table of references

References

USB-IF. (2017). Universal Serial Bus Power Delivery Specification rev 3.0. USB-IF.

Revision history

Table 9. Document revision history

Date	Revision	Changes
20-Feb-2019	1	Initial version
9-Apr-2019	2	Updated: *Figure 13*, *Figure 14* and *Figure 20* Unchunked Mode (UNCKD) replaces Fast Role Swap (FRS)