如何使用STM32CubeMX构建简单的USB-PD接收器应用程序

介绍

本应用笔记是从STM32CubeMX开始构建一个非常简单的USB供电接收器示例的指南。本文档适用于所有嵌入UCPD（USBType-C®Power Delivery控制器）外设的STM32 MCU。

不同屏幕截图中使用的主要硬件基于STM32G0系列微控制器，其相关固件包含在STM32CubeG0 MCU软件包中，但此文档中添加了一些注释，因此也可以使用STM32GubeG4系列及其相关固件的STM32G4系列微控制器。 。 STM32L5系列微控制器的关联固件在STM32CubeL5中。

X-NUCLEO-USBPDM1或X-NUCLEO-SNK1M1屏蔽罩关联TCPP01　M12保护电路，并提供USBType-C®连接器。 STM32L5 Nucleo　144，发现板和评估板在嵌入TCPP01　M12芯片时已准备好使用USB-PD。

本文档详细介绍了如何使用图1和图2所示的两种屏蔽版本构建USBPD接收器应用程序。

X-NUCLEO-SNK1M1的一些跳线已由焊桥取代，并且默认情况下，只有一个附加功能未连接，即SINK 5V至3A，无需USB供电。

Figure 1. STM32G0 Nucleo-64 board equipped with X-NUCLEO-USBPDM1 shield

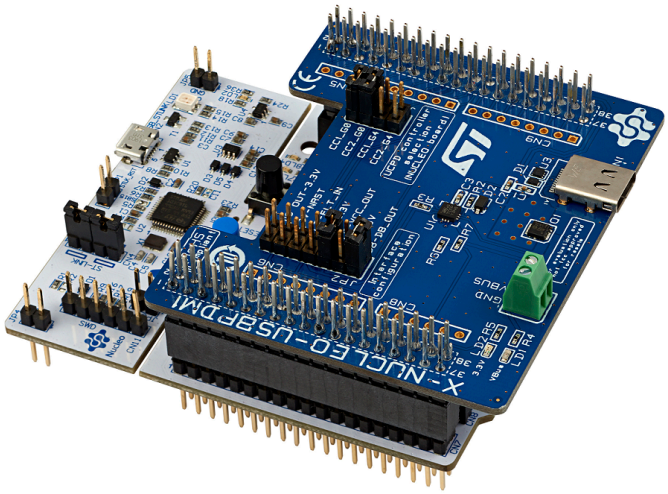
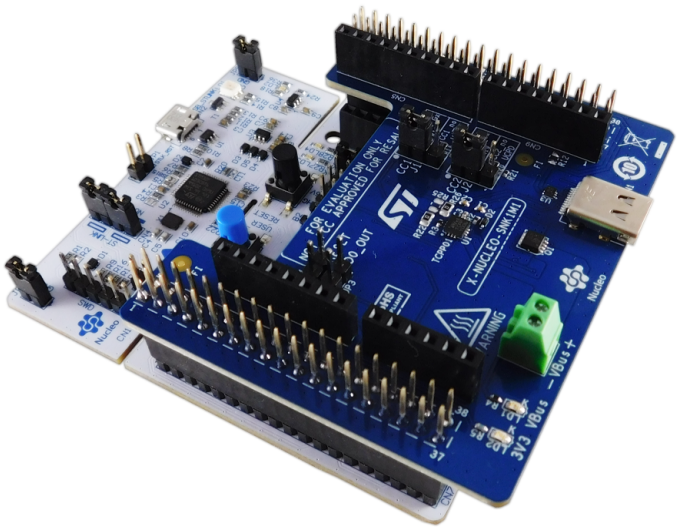


Figure 2. STM32G0 Nucleo-64 board equipped with X-NUCLEO-SNK1M1 shield



Pictures are not contractual.

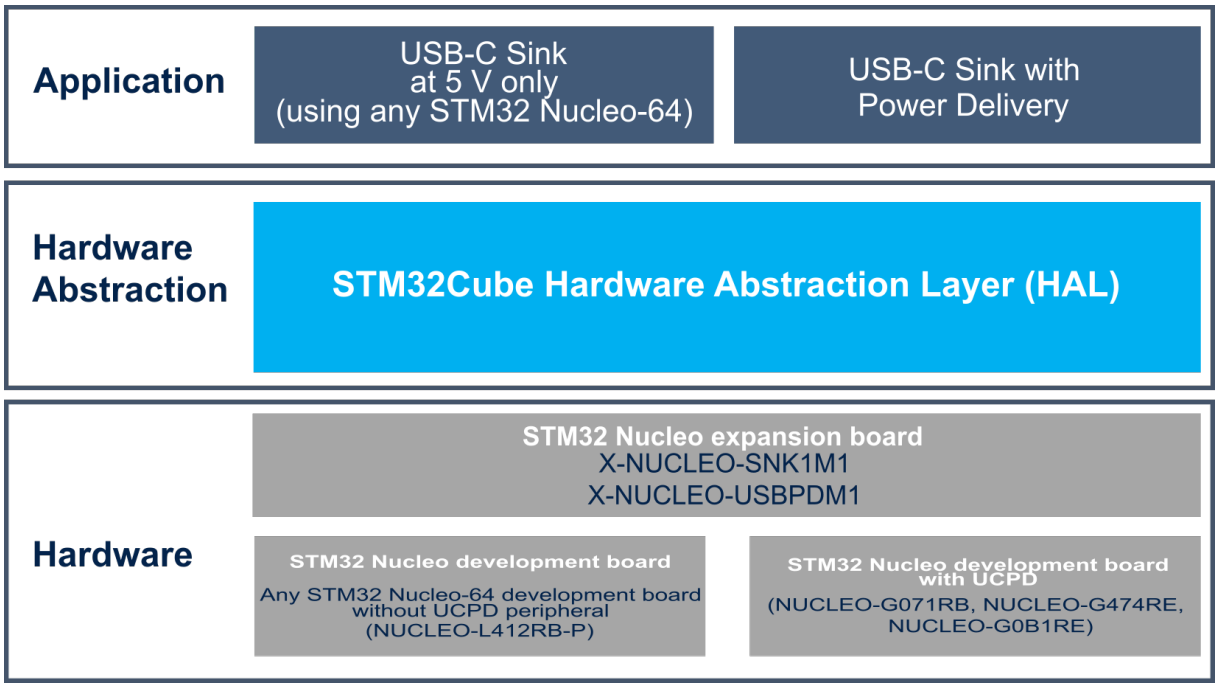
1 General information

简单的USB-PD接收器应用程序可在基于Arm®Cortex®-M处理器的STM32G0系列，STM32G4系列和STM32L5系列32位微控制器上运行。

Note: Arm是Arm Limited（或其子公司）在美国和/或其他地方的注册商标。

arm

Figure 3. **X-CUBE-TCPP框图架构**



即使本应用笔记以创建USB-PD应用程序为目标，TCPP01　M12屏蔽也可用于Type-C应用程序，如左上角的应用程序框中所示的体系结构所述。 仅C型应用程序不在此处描述。 对于信号源（TCPP02-M18）或DRP（TCPP03-M20）应用，还可以使用其他屏蔽，本文档也未介绍。

2 Acronyms

Table 1. **Acronym definitions**

Acronym	Definition
AMS	Atomic message sequence
APDO	Augmented power delivery object. Specific PDO to handle PPS
BSP	Board support package
CAD	Cable detection module responsible for attaching or detaching detection
CC	Communication channel
DPM	Device policy manager. In the power delivery context, this part corresponds to the user application.
DRP	Dual role power. The ability for a product to either source or sink power.
GUI	Graphical user interface. It applies here to STM32CubeMonitor-UCPD (STM32CubeMonUCPD).
HAL	Hardware abstraction layer
HW	Hardware
LL	Low layer
OVP	Over-voltage protection
PDO	Power delivery object
PPS	Programmable power supply
PE	Policy engine
SNK	Power sink. Ability to request power
SRC	Power source. Ability to provide power
TCPP	Type-C port protection
UCPD	USB Type-C® power delivery. A new peripheral in some STM32 series, like STM32G0 Series, STM32G4 Series or STM32L5 Series, which manages power delivery protocol communication with two lines.

3 参考文件

STMicroelectronics ecosystem material:

Name	Title/description
STM32CubeMX	STM32CubeMX: STM32Cube initialization code generator
UM2552	用户手册使用STM32微控制器管理USB供电系统（UM2552）
UM2468	用户手册STM32CubeMonitor-UCPD软件工具，用于USB Type-C™供电端口管理（UM2468）
DS12900	TCPP01　M12数据表VBUS和CC线路的USB-C过压保护（DS12900）
AN4871	应用笔记USB Type-C保护和滤波（AN4871）
DB3747	Databrief STM32CubeMonitor-UCPD软件工具，用于USB Type-C™供电端口管理（DB3747）
TA0357	技术文章USB Type-C和电源传输技术概述（TA0357）
AN5225	应用笔记，使用STM32 MCU和MPU的USB Type-C供电（AN5225）
[YouTube_video]	YouTube视频STM32G0：在不到10分钟的时间内创建USB Power Delivery Sink应用程序
UM2773	用户手册基于STM32 Nucleo的基于TCPP01-M12的X-NUCLEO-SNK1M1 USB Type-C™电力输出接收器扩展板（UM2773）
[USBPD_overview_wiki]	USBPD overview wiki

USB specification documents:

Name	Title/description
[USB2.0 specification]	USB2.0 Universal Serial Bus Revision 2.0 Specification
[USB3.1 specification]	USB3.1 Universal Serial Bus Revision 3.2 Specification
[USB battery charging specification]	USB BC Battery Charging Specification Revision 1.2
[USB Type-C® cable and connector specification]	Universal Serial Bus Type-C Cable and Connector Specification 2.0, August 2019
[USB-PD specification]	Universal Serial Bus Power Delivery Specification, Revision 3.0, Version 2.0, August 28, 2019

4 入门

目的是为UCPD外设配置一个USB-PD堆栈，并检查是否已达成第一个合同，这意味着接收器找到了匹配的电源。因此，还需要任何壁式充电器或任何经过电源传送认证的电源。
为了实现此目标，必须执行以下步骤：

1.设置UCPD外设，以使用STM32CubeMX在CC线上暴露Rd电阻，并从源中检测Rp。
2.从随附的源中读取VBUS。 初始化部分由STM32CubeMX完成，但是必须在应用程序中手动添加测量开始。
3.最后，向源发送电力输送请求消息，并达成明确的合同。 在由STM32CubeMX生成应用程序文件之后，只能通过编辑应用程序文件来手动完成此操作。
本文档中介绍了可选步骤，以帮助用户进行调试：
4.添加跟踪实用程序，该实用程序使用ST-LINK虚拟COM端口从板上获取一些调试信息
5.添加了嵌入式工具以与STM32CubeMonitor-UCPD通信，这是一个Java应用程序（GUI），可帮助构建该应用程序

5 STM32CubeMX分步顺序

5.1 强制性零件

由于某些STM32器件中存在UCPD外设，例如STM32G071xx（或带有USB支持的新STM32G0）和STM32G474xx，因此可以执行以下步骤。 该外设管理CC线路上的电源传输通信。 X-NUCLEO-USBPDM1或X-NUCLEO-SNK1M1屏蔽层嵌入TCPP01　M12单芯片解决方案，在CC线上增加了微控制器保护，例如ESD，过压和过热，并且还 可以访问USBType-C®插座。

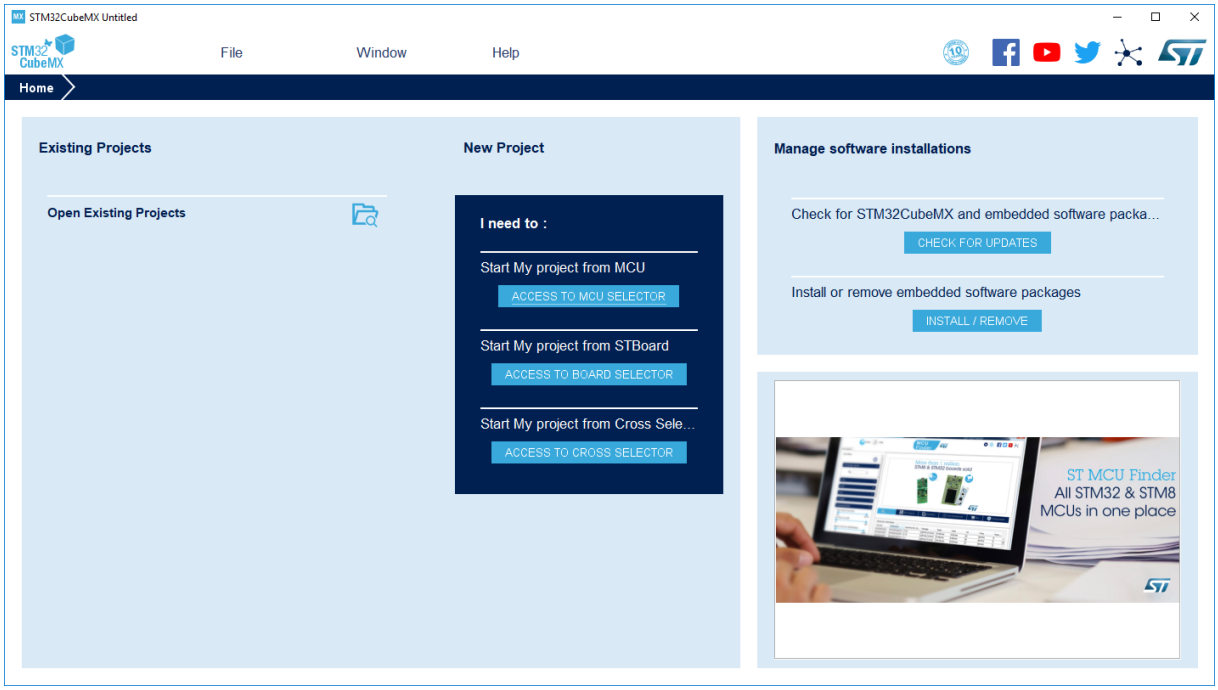
TCPP01　M12已经嵌入STM32L5 Nucleo-144，发现和评估板中。

在开始之前，请先检查是否同时使用了最新版本的STM32CubeMX和STM32CubeG0，STM32CubeG4或STM32CubeL5 MCU软件包。

以下小节中基于带有TCPP01　M12防护罩的NUCLEO-G071RB STM32G0 Nucleo-64板描述的序列适用于其他配置。

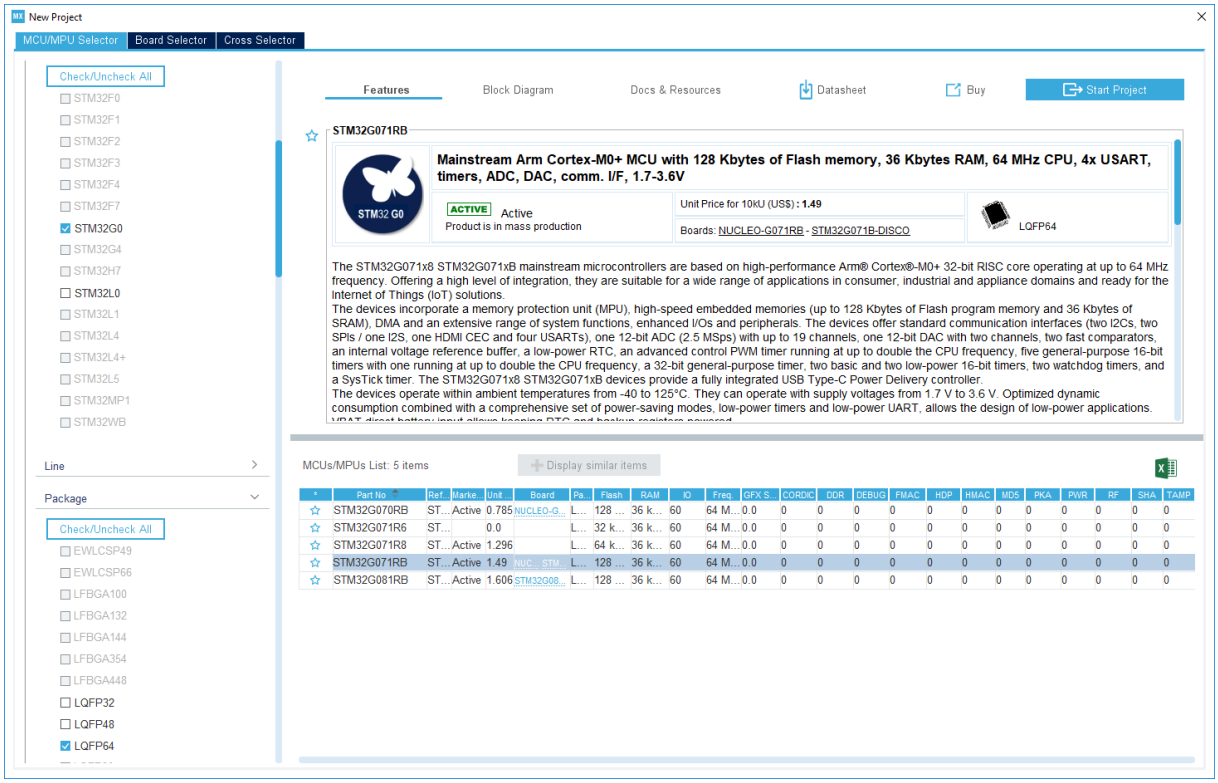
5.1.1 启动STM32CubeMX并选择MCU

Figure 4. Start STM32CubeMX



Create a new project *File/New Project* or click on *ACCESS TO MCU SELECTOR*, and check *STM32G0* and *LQFP64 package*, to filter available MCUs, double click on *STM32G071RB*.

Figure 5. **Select the STM32G0 MCU**



5.1.2 UCPD外设配置

现在可以激活微控制器内部的UCPD外设。

该外设管理电源输送检测及其通过CC线路的通信。 有关更多详细信息，请参见参考手册STM32G0x1基于Arm®的高级32位MCU（RM0444）的第35节USB Type-C™/ USB供电接口（UCPD）。

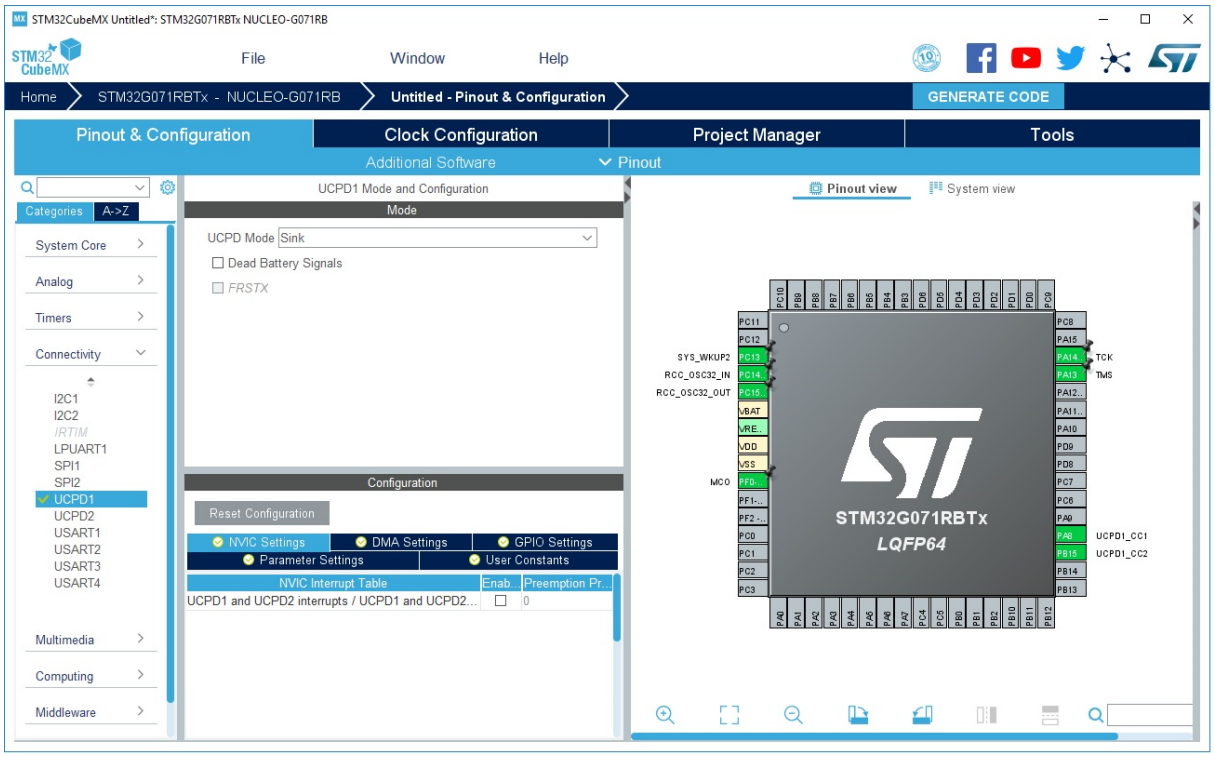
单击“连接性”，然后选择“ UCPD1”。 STM32G0具有UCPD模块的两个实例。 TCPP01 M12屏蔽层已连接到UCPD实例1。可以使用实例2，而不是将屏蔽层插入Nucleo板上，而是使用飞线。

该演示运行一个接收器应用程序。 因此，对于UCPD模式，用户选择接收器。 解开死者

电池信号以避免使用UCPD外设的内部电池耗尽管理，因为在演示中，ST-LINK为套件（Nucleo和屏蔽）供电，并且绕过了TCPP01 M12的电池耗尽管理，因为 从简单的应用程序开始，DB-3.3V跳线已打开。

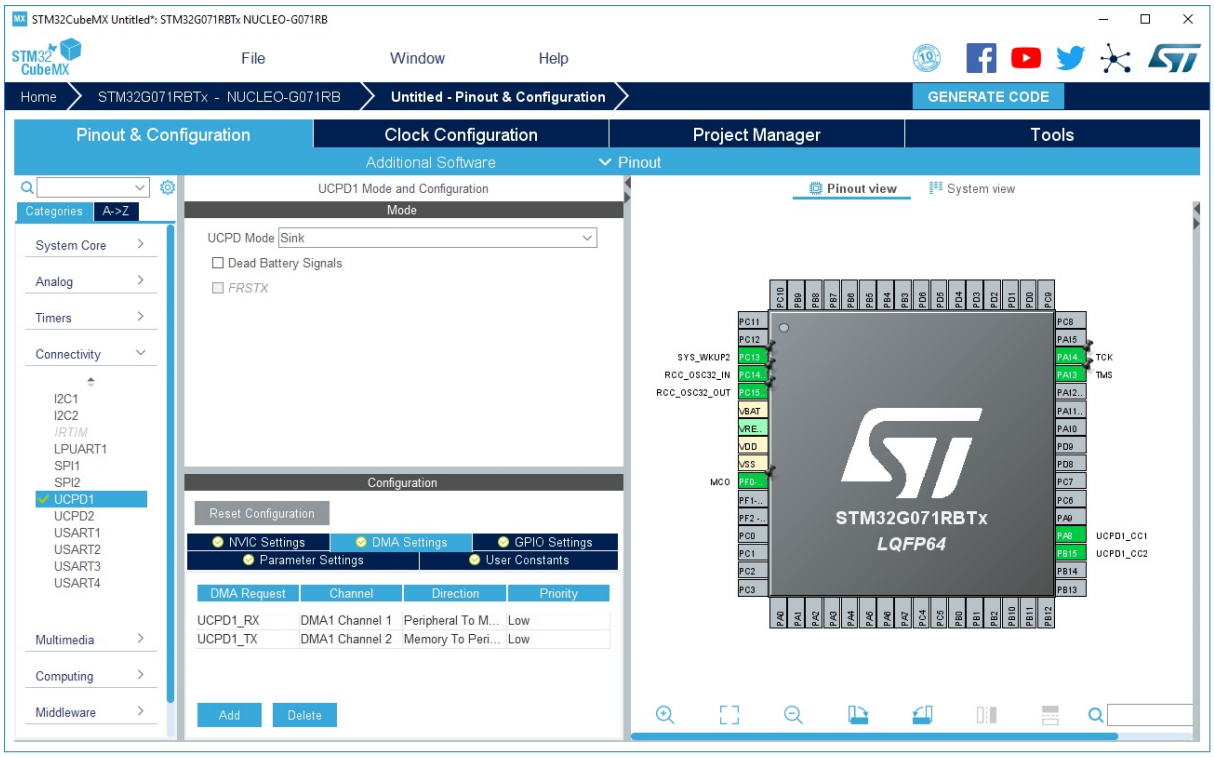
Note: *第5.7节中详细介绍了电量耗尽的电池的屏蔽跳线配置*

Figure 6. **UCPD外设基本配置**



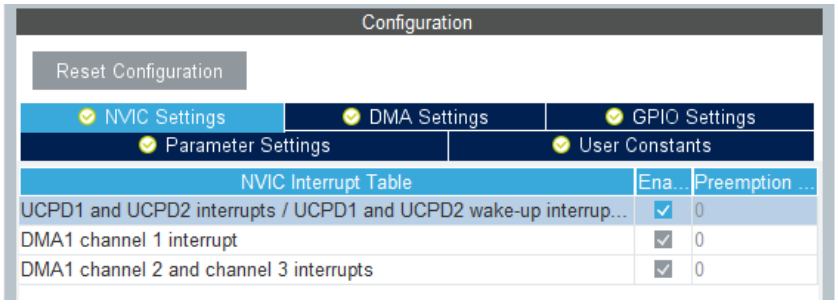
现在，必须添加用于TX和RX路径的DMA，并启用中断。

Figure 7. **UCPD外设DMA配置**



我们的设备需要DMA初始化才能通过UCPD外设进行PD通信，因此用户必须为TX和RX配置DMA通道。 例如，在DMA1通道1上设置RX，在DMA1通道2上设置TX。

Figure 8. UCPD外围IT激活



启用了两个DMA处理程序，但固件未直接使用它们。 所有的UCPD处理都是通过UCPD处理程序完成的。

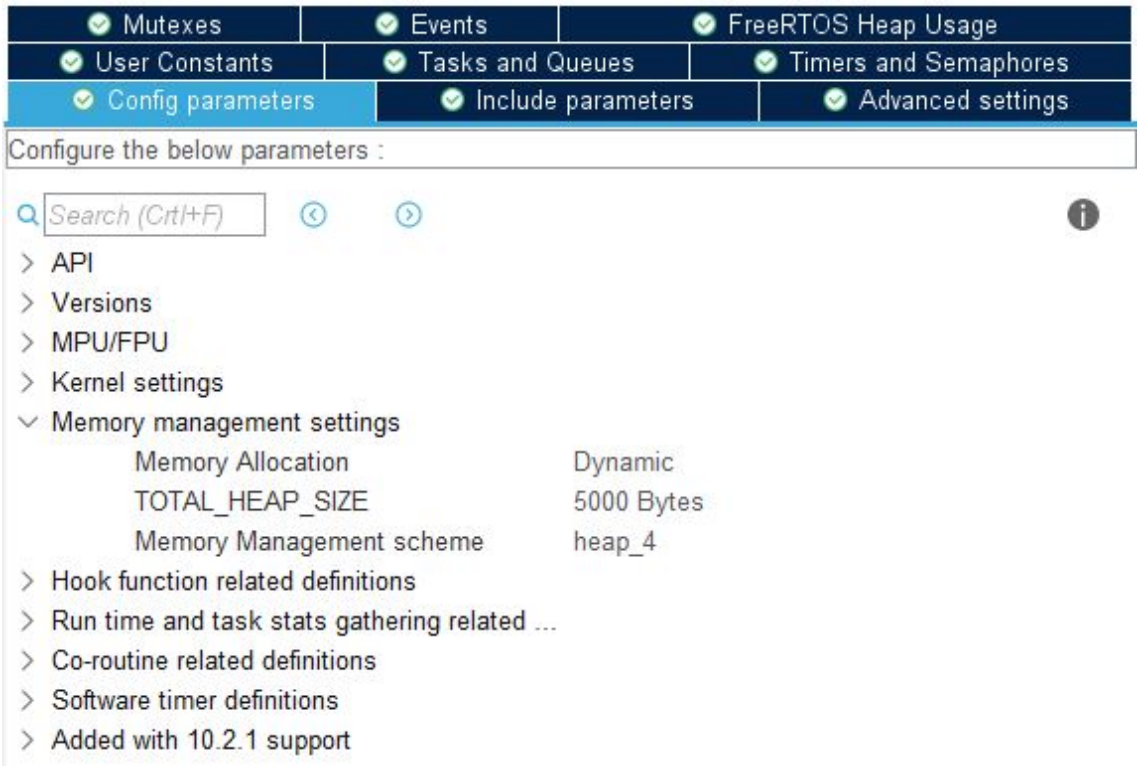
5.1.3 FreeRTOS™配置

在中间件类别中，启用FreeRTOS™。

选择CMSIS_V1并将TOTAL_HEAP_SIZE设置为5000。从STM32L5和其他固件包交付中，必须选择CMSIS_V2而不是CMSIS_V1。

这里的堆大小是一个开始。 在优化最终应用程序时，必须稍后对其进行调整。

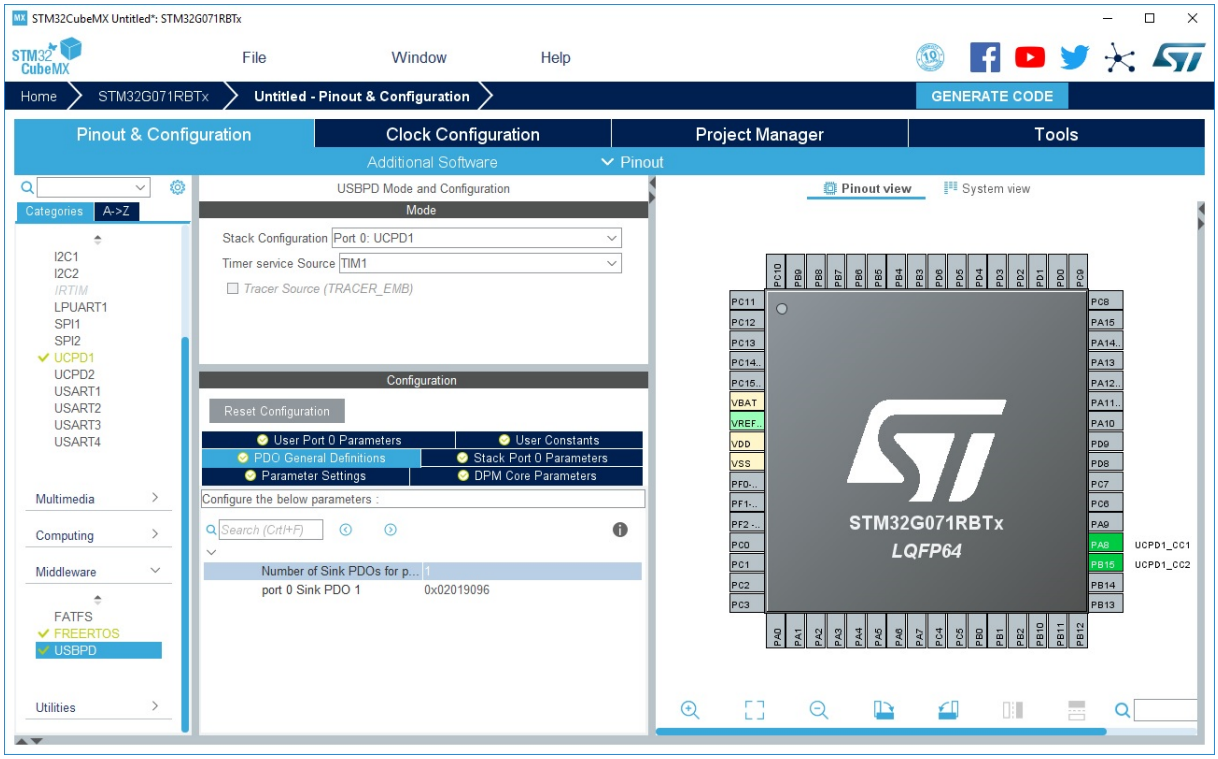
Figure 9. FreeRTOS™配置



5.1.4 USB-PD中间件配置

在中间件类别中，选择USB-PD，然后在USBPD模式和配置内的下拉列表中选择端口0：UCPD1。 有关整个固件中USB-PD堆栈角色的更多详细信息，请参考UM2552。

Figure 10. USB-PD中间件配置



检查端口0接收器PDO1是否设置为0x02019096，即使它并不重要，因为它在当前未使用简单的例子。 0x02019096表示5000 mV，1500 mA，双角色数据，无快速角色交换。 有关更多信息，请参考[USB-PD specification(USB-PD规范)]中的表6-14，该图在图11中进行了复制。

Figure 11. 规范详细信息（通用串行总线电源传输规范中的表6-14)

Table 6-14 Fixed Supply PDO - Sink

Bit(s)	Description
B31...30	Fixed supply
B29	Dual-Role Power
B28	Higher Capability
B27	Unconstrained Power
B26	USB Communications Capable
B25	Dual-Role Data
B24...23	Fast Role Swap required USB Type-C Current (see also *[USB Type-C 1.3]*): **Value** / **Description** 00b – Fast Swap not supported (default) 01b – Default USB Power 10b – 1.5A @ 5V 11b – 3.0A @ 5V
B22...20	***Reserved – Shall*** be set to zero.
B19...10	Voltage in 50mV units
B9...0	Operational Current in 10mA units

图12说明了应用值0x02019096。

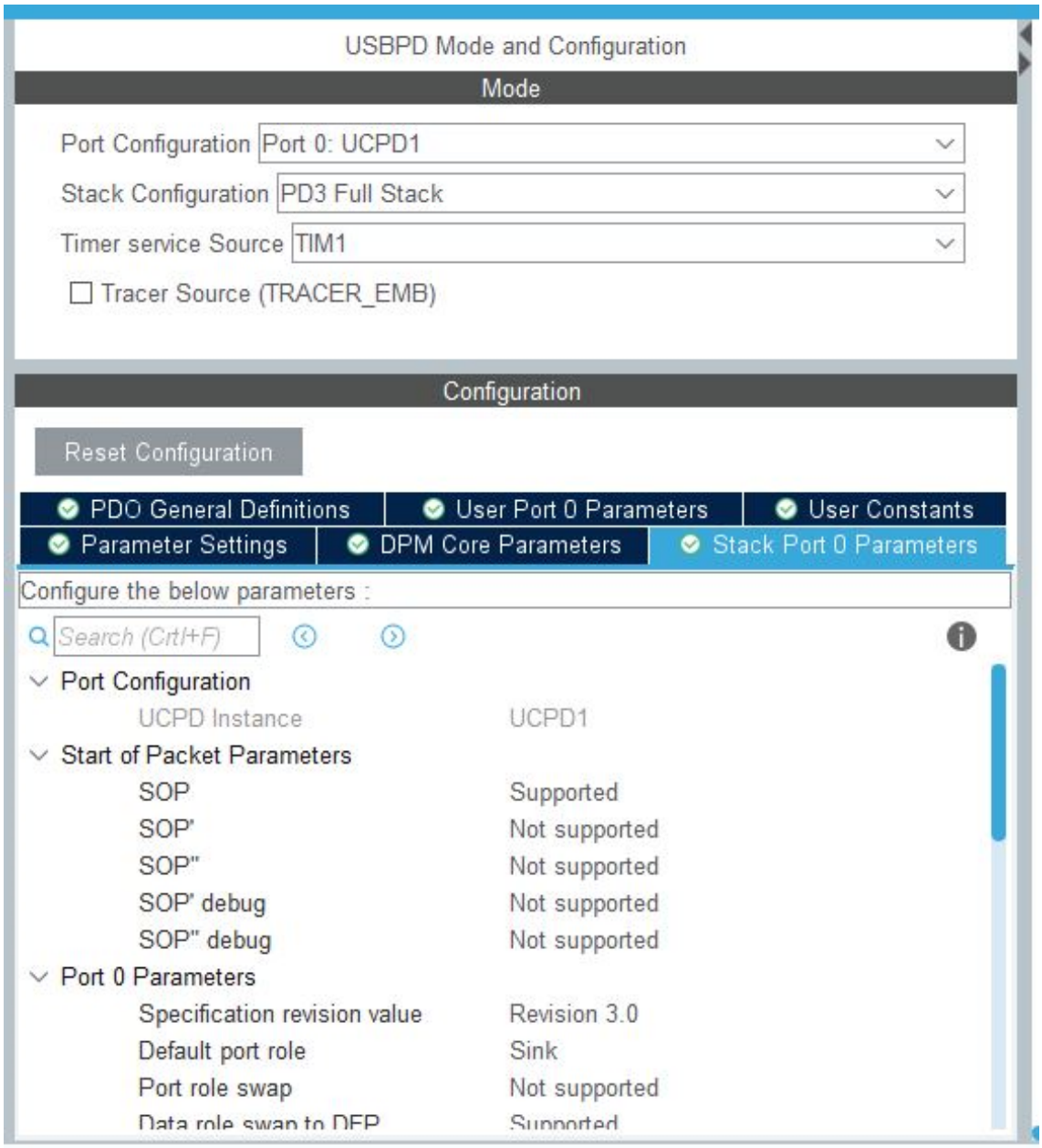
Figure 12. 详细的PDO解码

0x02019096 = **00** 0 0 0 0 1 **00** 000 **0001100100** 0010010110b

Bit(s)	Description	Applied value	The definitions can be found in STM32 CubeG0 Firmware\Projects\STM32G081B-EVAL\Demonstrations\DemoUCPD\Inc\usbpd_pdo_defs.h
[31,30]	**Fixed supply**	**00b**	**USBPD_PDO_TYPE_FIXED**
[29]	Dual-role power	0b	USBPD_PDO_SNK_FIXED_DRP_NOT_SUPPORTED
[28]	**Higher capability**	**0b**	**USBPD_PDO_SNK_FIXED_HIGHERCAPAB_NOT_SUPPORTED**
[27]	Unconstrained power	0b	USBPD_PDO_SNK_FIXED_EXT_POWER_NOT_AVAILABLE
[26]	**USB communications capable**	**0b**	**USBPD_PDO_SNK_FIXED_USBCOMM_NOT_SUPPORTED**
[25]	Dual-role data	1b	USBPD_PDO_SNK_FIXED_DRD_SUPPORTED
[24,23]	**Value** / **Description** **00b** **Fast swap not supported (default)** 01b Default USB power 10b 1.5 A at 5 V 11b 3.0 A at 5 V	**00b**	**USBPD_PDO_SNK_FIXED_FRS_NOT_SUPPORTED**
[22-20]	Reserved – Must be set to zero.	000b	
[19-10]	**Voltage in 50 mV units**	**0001100100b = 0x64**	**0x64*50 = 5000 mV**
[9-0]	Operational current in 10 mA units	0010010110b = 0x96	0x96*10 = 1500 mA

有关PDO定义的更多详细信息，请查看UM2552中的POWER_IF部分。

Figure 13. 详细的堆栈配置

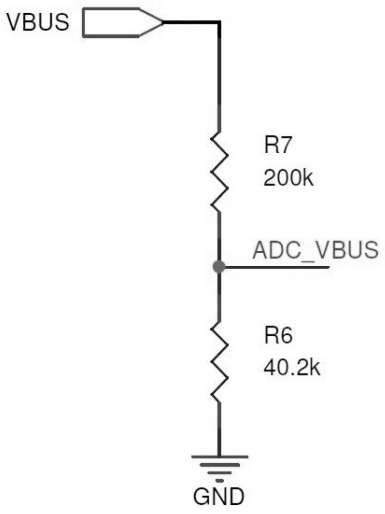


在“堆栈端口0参数”选项卡中，用户在其他参数中选择他要支持的内容，例如PD3.0规范修订版。 这些参数均为功率传输设置。 无需为第一个应用程序更改它们。 有关更多信息，请参阅[USB-PD规格]。

5.1.5 用于VBUS读取的ADC配置

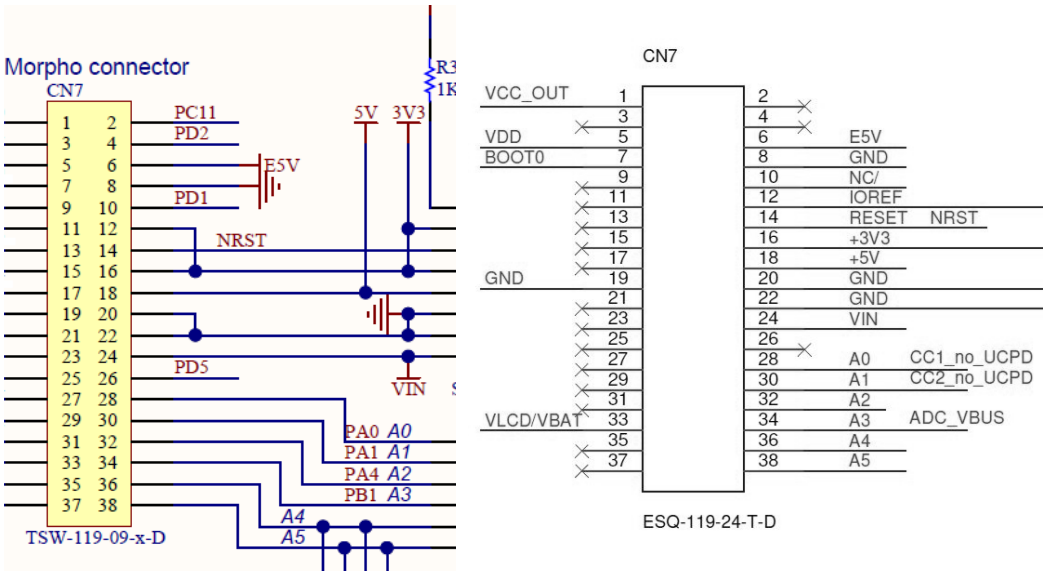
为了遵守Type-C状态机，必须进行VBUS检测。 为此，我们使用连接到分压器桥的ADC，以保持在GPIO STM32电压范围内。

Figure 14. TCPP01-M12 shield voltage divider



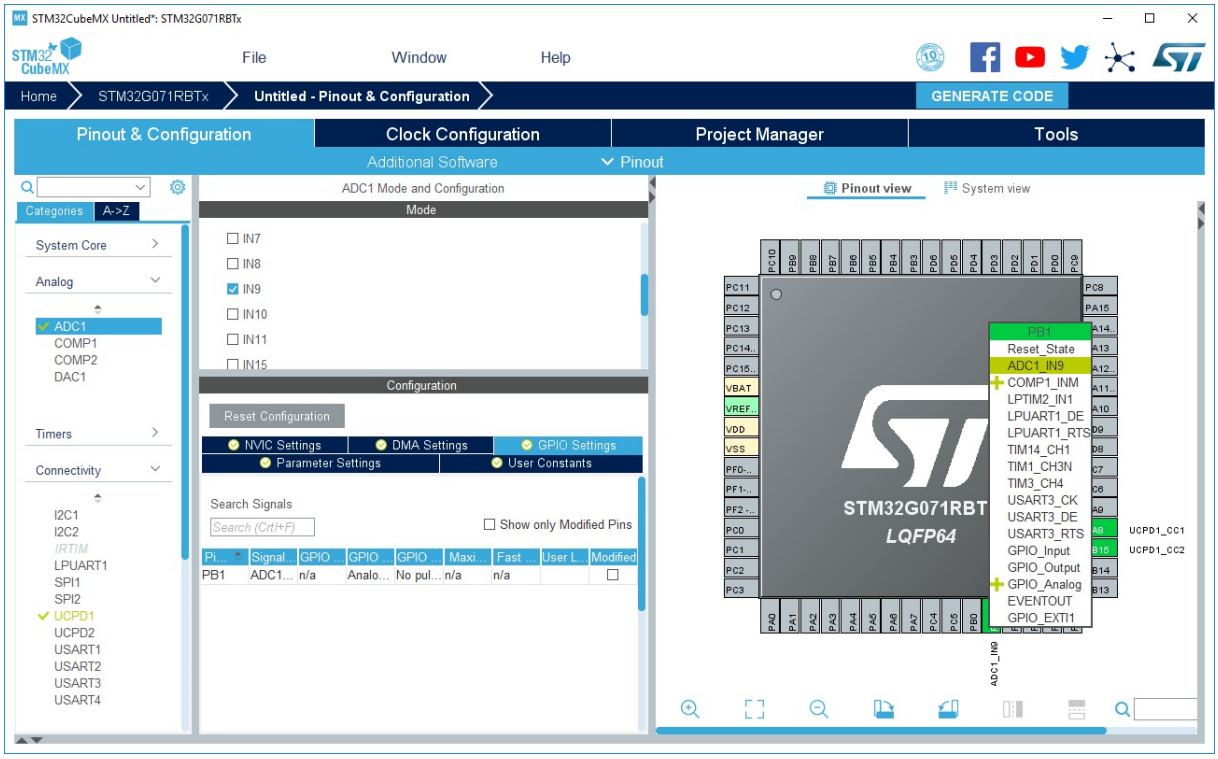
在CAD资源下的原理图包中查看X-NUCLEO-SNK1M1屏蔽原理图，VBUS在CN7 ST morpho连接器插针34上，对应于PB1，如图15所示。

Figure 15. STM32G0 Nucleo-64开发板（左）和TCPP01　M12屏蔽（右）原理图



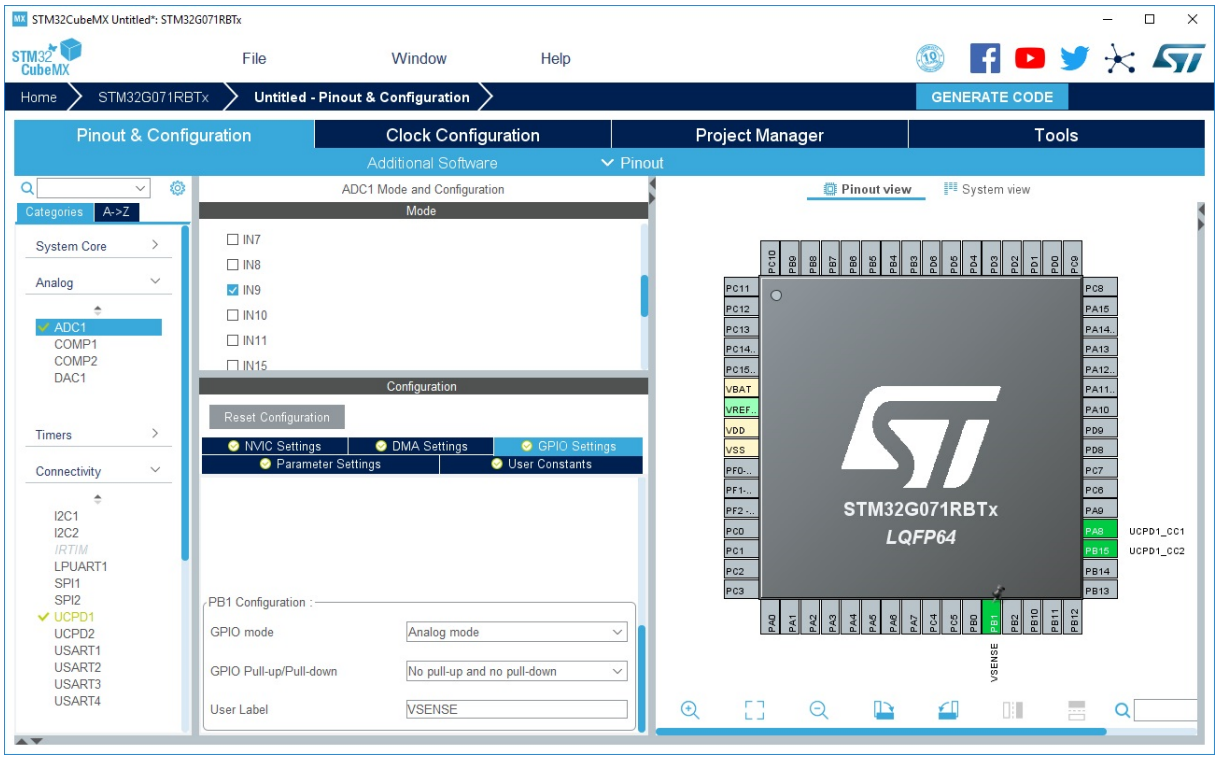
在模拟类别中，选择ADC1。
单击STM32CubeMX中的右侧，PB1连接到ADC1_IN9备用功能输入。 因此，请在STM32CubeMX窗口的模式部分中选择它，或在引脚视图中选择ADC1_IN9。

图16. ADC配置



然后在GPIO设置选项卡中，为此信号添加用户标签VSENSE：

图17. ADC GPIO设置

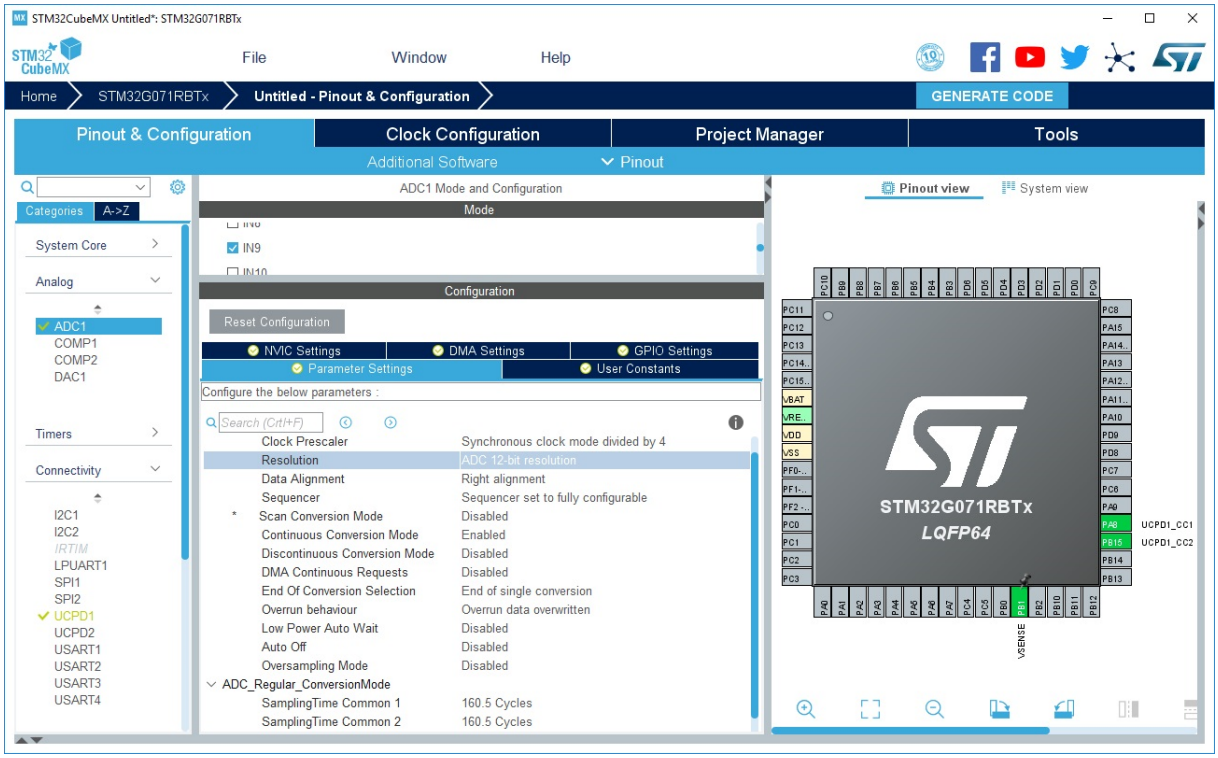


选择ADC1的简单和基本设置：

- 时钟预分频器：同步模式除以4
- 连续转换模式：已启用
- 溢出行为：溢出数据被覆盖
- 采样时间：160.5个周期

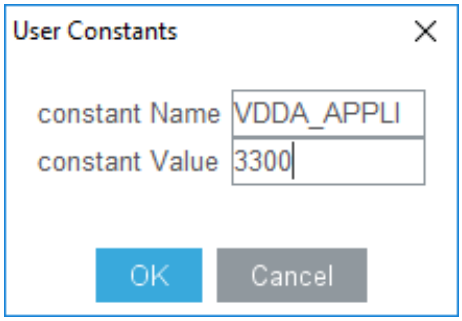
必须通过与测量值相关的阻抗来调整采样时间。 如果是X-NUCLEO-USBPDM1 TCPP01　M12屏蔽层，电阻大于10KΩ，因此，优选高循环次数。

Figure 18. ADC parameters settings



ADC的最新版本：创建了具有3300值的用户常数VDDA_APPLI，代表3.3 V的ADC参考电压。此变量由生成的代码调用。

图19. ADC用户常数



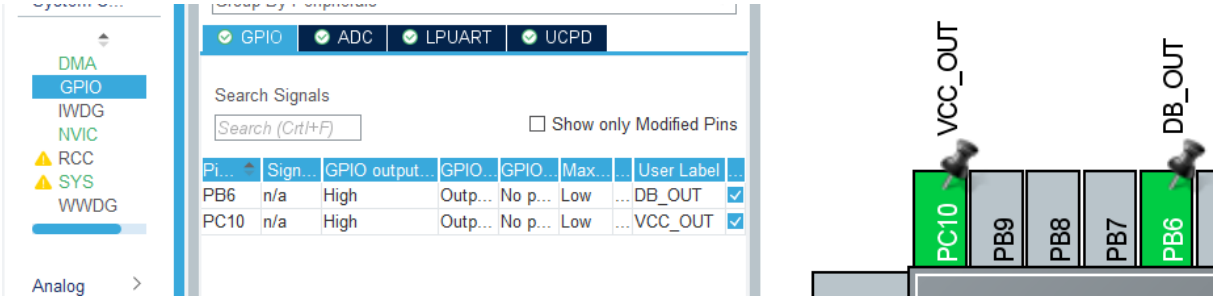
Note: *在STM32G4 Nucleo-64板上，由于与STM32G0的ADC映射差异，必须使用ADC IN15信号代替IN9，并且等级采样时间可以设置为247.5个周期。*

5.1.6 额外的GPIO设置

对于X-NUCLEO-SNK1M1屏蔽，需要两个附加的GPIO设置（在X-NUCLEO-USBPDM1中不需要，因为设置是由跳线强制执行的），如图20所示。

1. PB6（禁用死电池的DB_OUT）GPIO输出为高电平
2. PC10（VCC_OUT引脚为TCPP01　M12上电）的GPIO输出为HIGH

Figure 20. Modify TCPP01-M12 controls

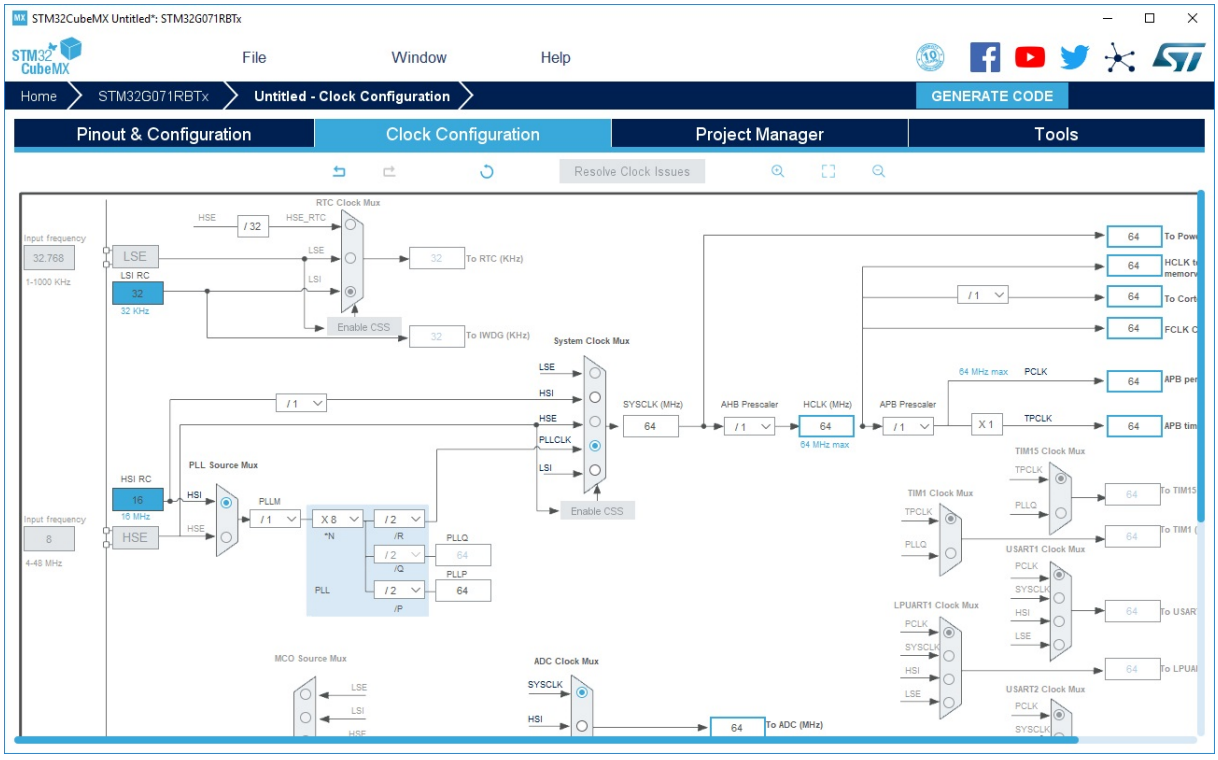


对于实际应用，必须在UCPD初始化之后执行这些GPIO设置。

5.1.7 时钟检查

可以选择PLLCLK作为系统时钟多路复用器的输入时钟，以产生最小设置为16 MHz的SYSCLK和HCLK。 没有最大限制。 对于STM32G4，它可以为170 MHz。 HSI用于为UCPD外设提供时钟，因此必须将其启用。

Figure 21. Clock configuration



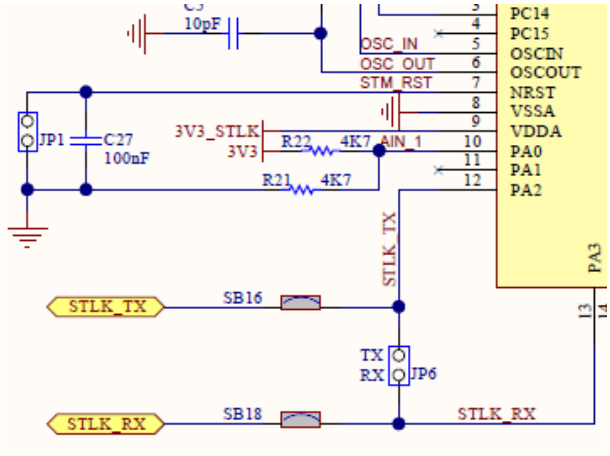
简单的USB-PD接收器应用程序的必需设置已完成。 强烈建议以下部分进行调试。

5.2 其他推荐的可选调试

5.2.1 用于调试的UART配置

在STM32G0 Nucleo-64板上，连接到ST-LINK的虚拟COM端口是LPUART1。

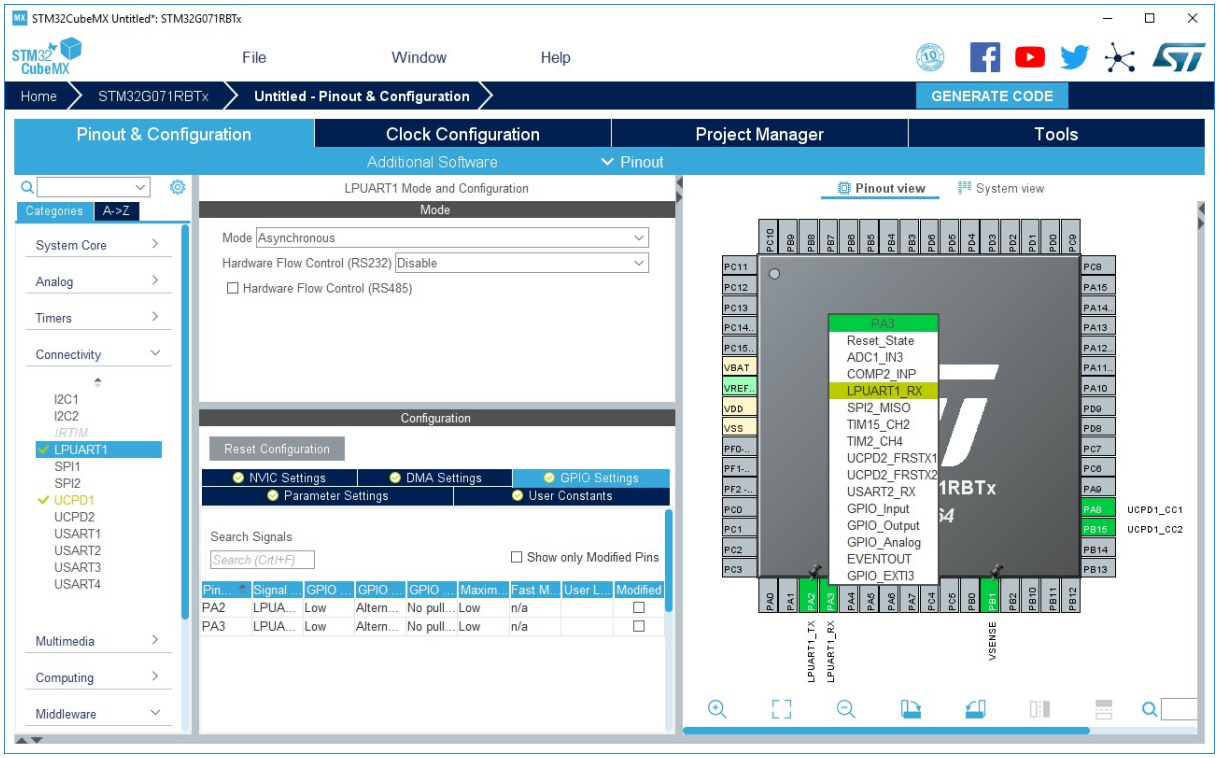
Figure 22. STM32G0 Nucleo-64 board STLK connection



该接口在STM32CubeMX中以异步模式激活。

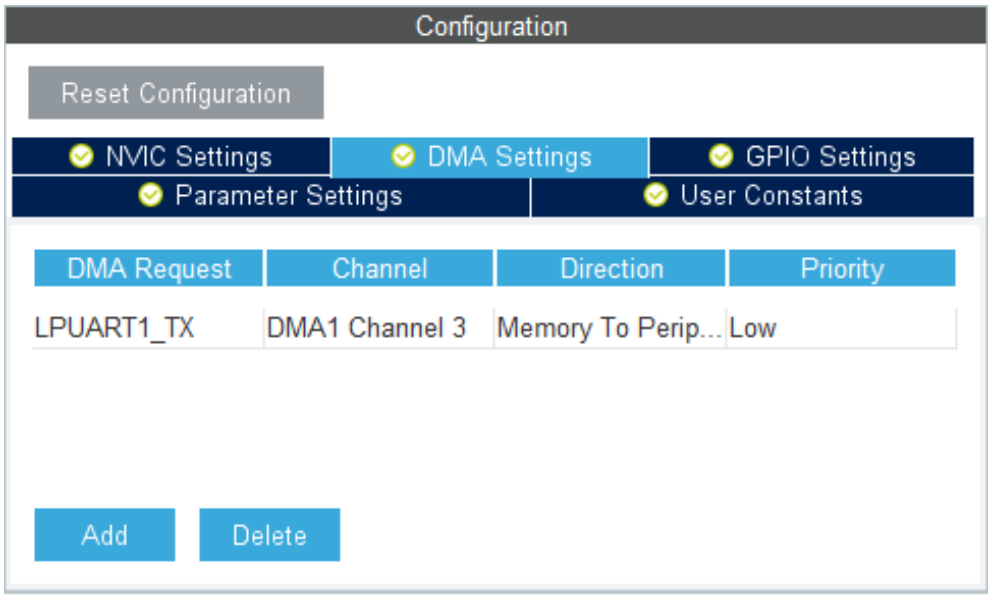
重要的：
必须更改LPUART1使用的默认STM32CubeMX引脚以匹配STM32G0 Nucleo-64硬件：用于TX的PA2和用于RX的PA3。

Figure 23. LPUART1 activation and GPIO pin (TX and RX) update



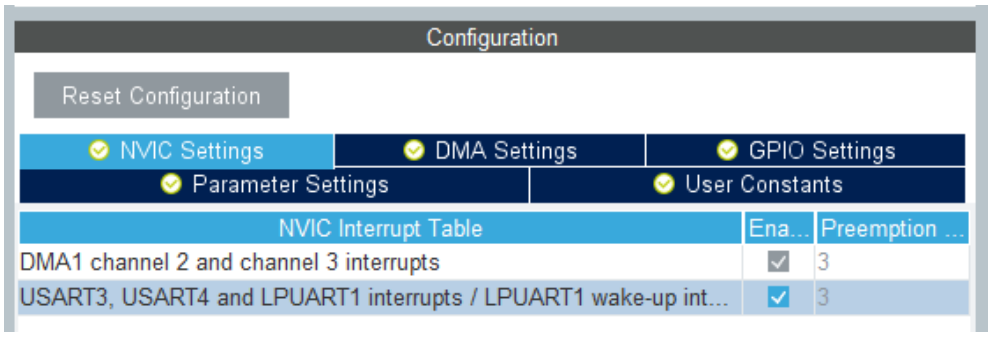
Then the DMA requests are activated for the TX path only: DMA1 channel 3.

Figure 24. DMA activation for LPUART



And the interrupt is activated:

Figure 25. DMA activation for LPUART1



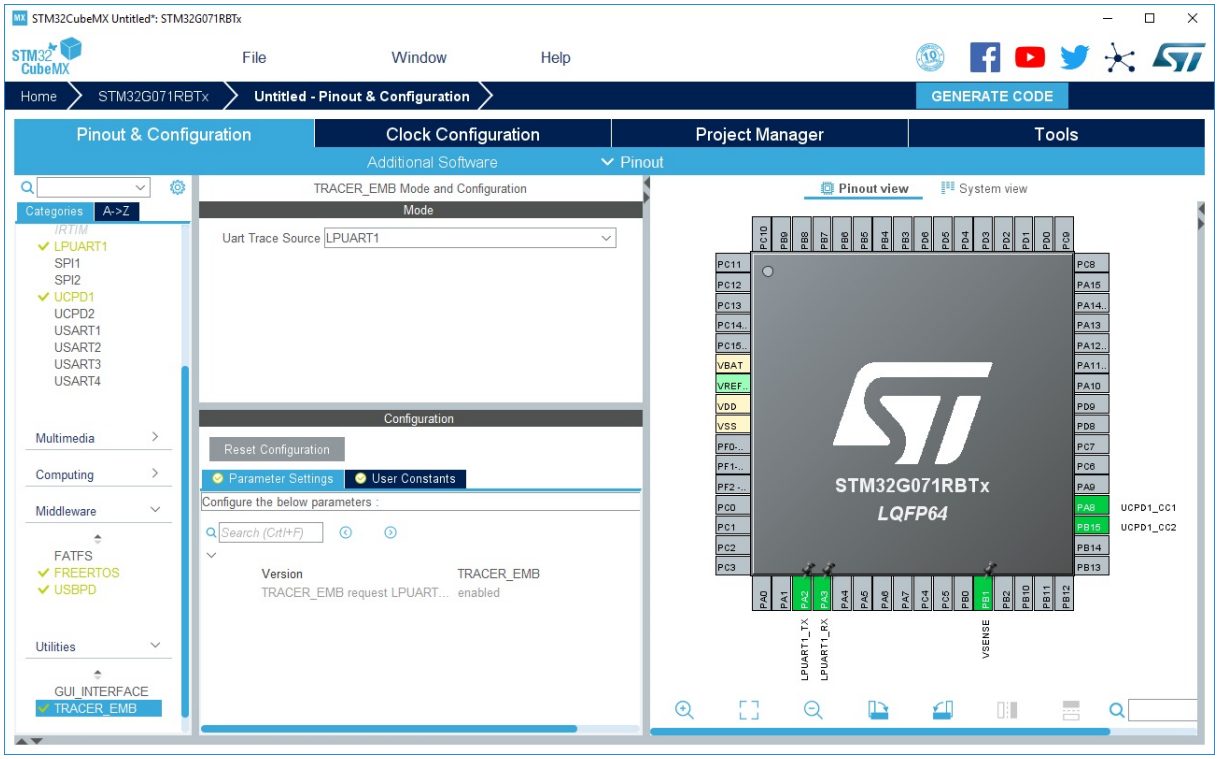
Note: If the STM32G4 Nucleo-64 board is used, USART1 must be used.

Note: The default UART configuration is used. The debug trace runs at 921600 bauds.

5.2.2 激活嵌入式跟踪器以进行调试

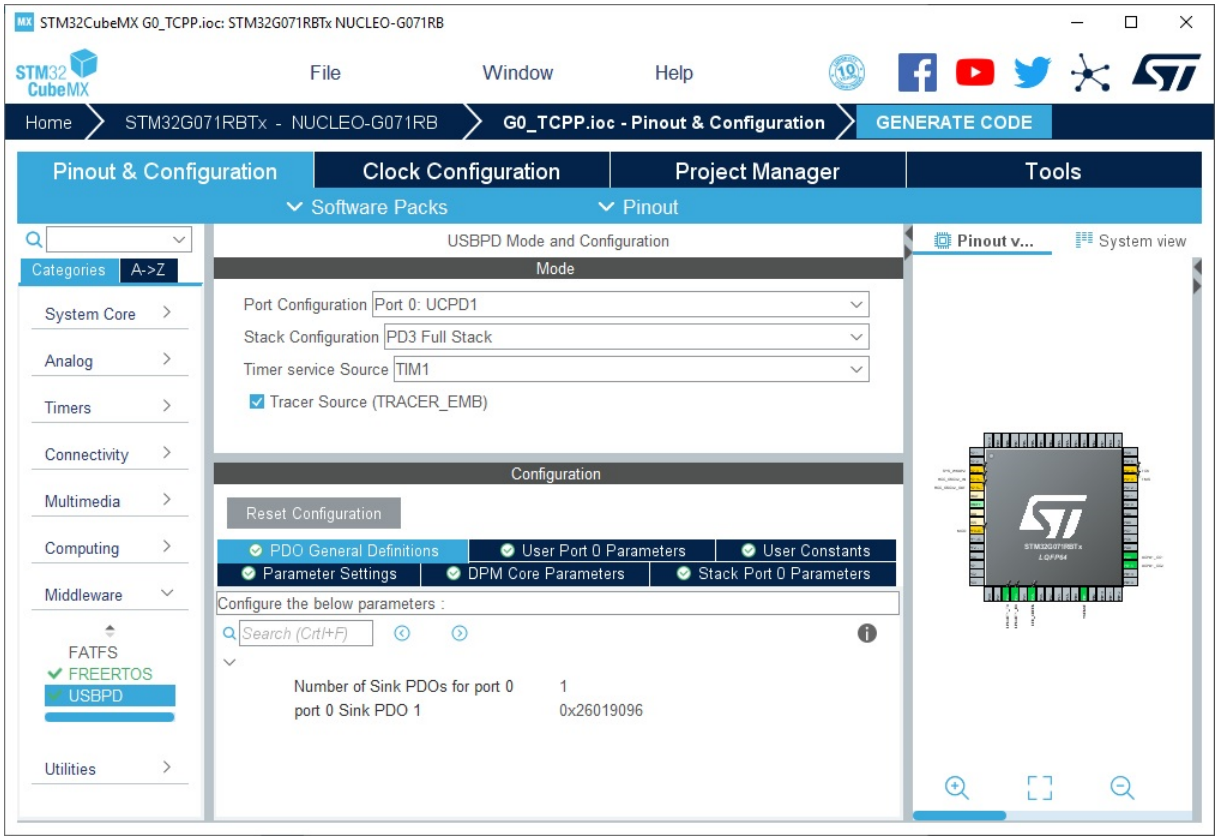
这是在实用程序类别中完成的：在UART跟踪源模式下，选择TRACER_EMB，然后选择LPUART1。

Figure 26. Activation of TRACER_EMB



回到USB-PD中间件配置，跟踪撤离已激活：检查TRACER_EMB的跟踪器源。

Figure 27. 选择USB-PD中间件TRACER_EMB源

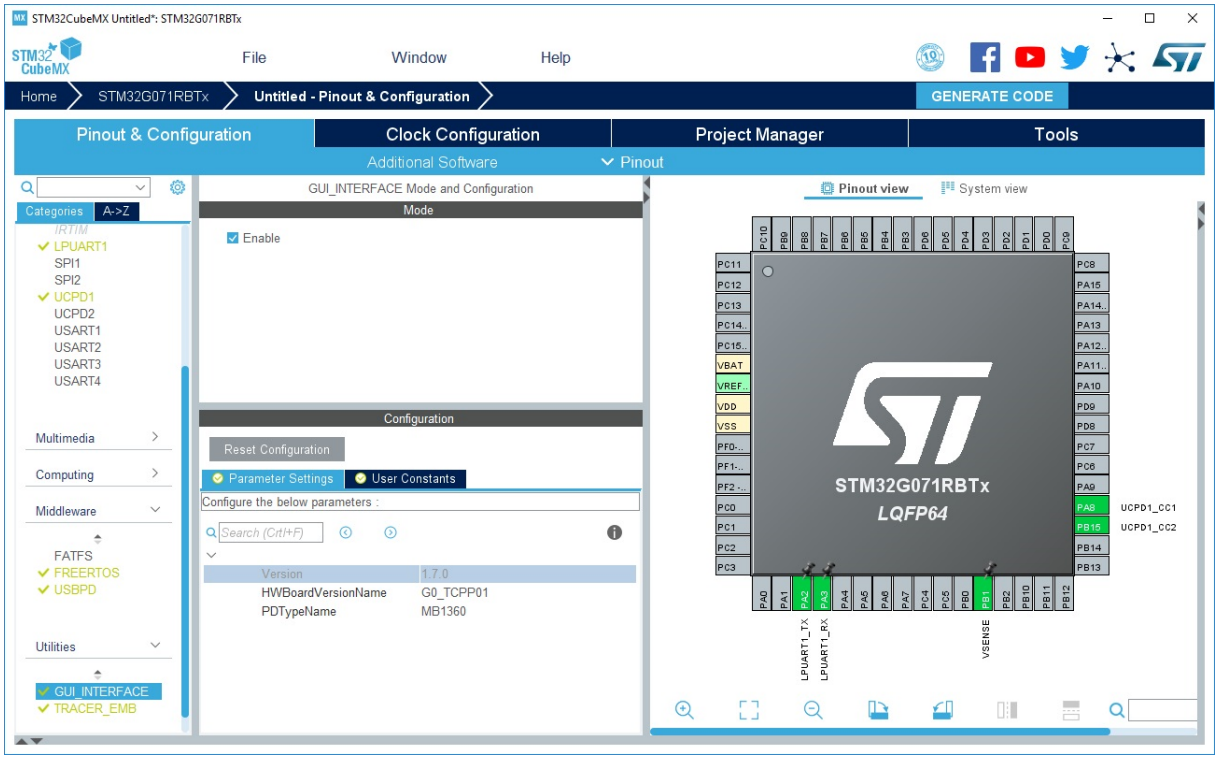


如果需要使用UCPD监视工具STM32CubeMonUCPD与USB-PD堆栈进行交互，则可以激活固件交互式堆栈响应器。

5.2.3 激活UCPD监视器固件响应器

可以在实用程序类别中激活该监视器：GUI_INTERFACE。 然后输入自由文本来描述板。

Figure 28. Activation of GUI_INTERFACE



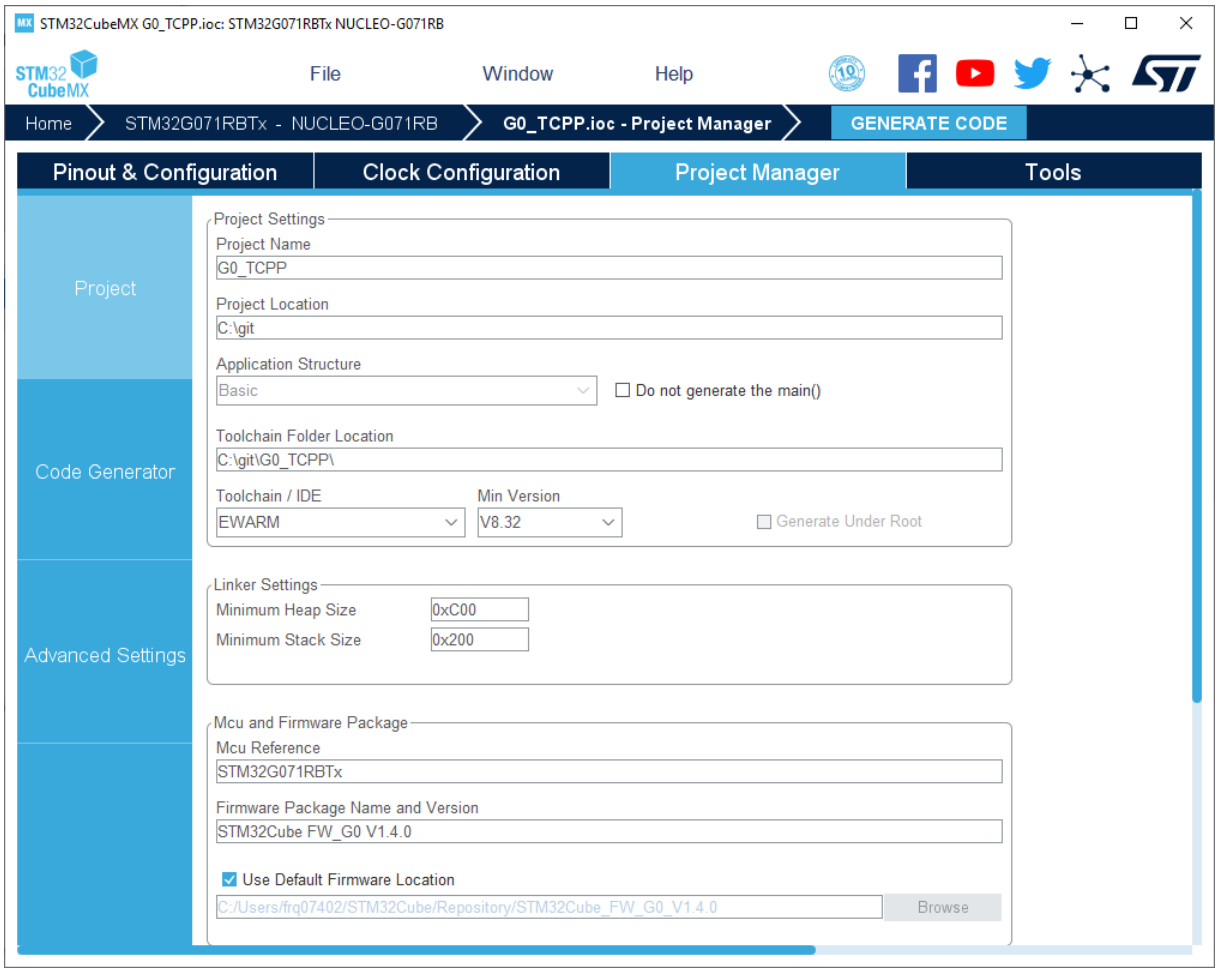
5.3 更新并保存项目配置

配置完成后，在保存项目之前，必须在项目管理器选项卡中保存很少的参数。

在项目管理器选项卡下，选择项目的名称。 对于项目目录，如果STM32CubeMX也不在一个驱动器中，请避免使用一个驱动器。

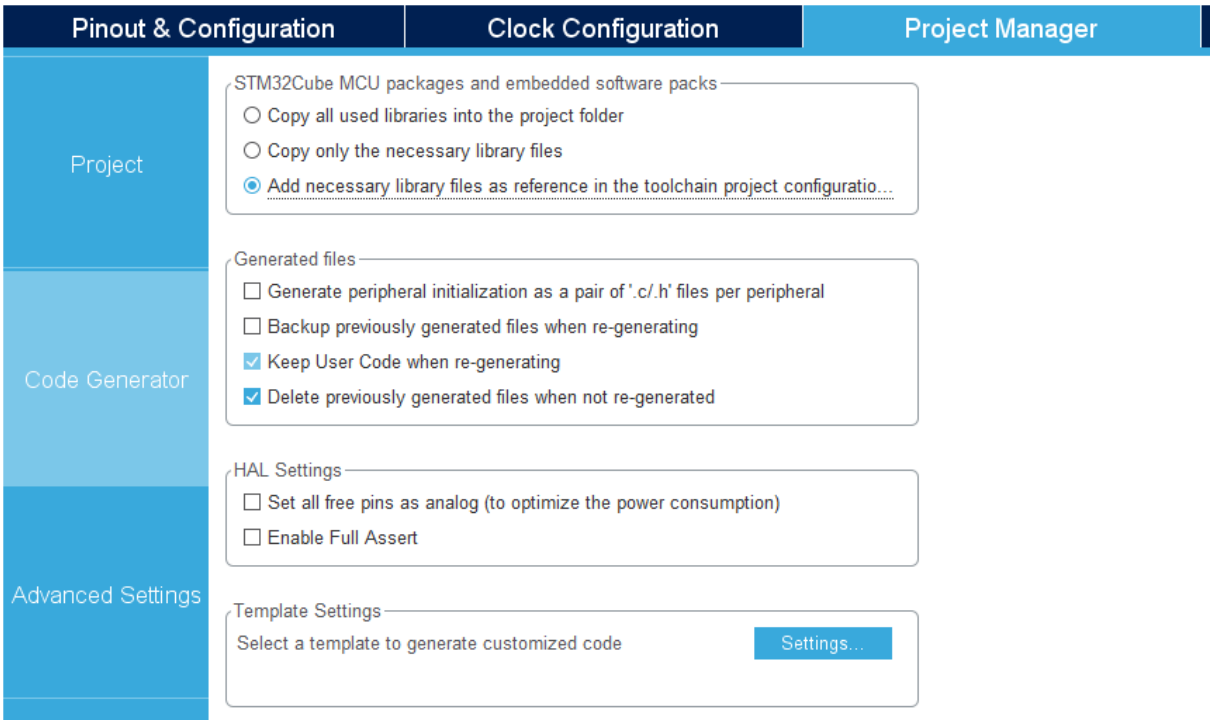
将最小堆栈大小配置为0xC00。 这是第一个版本，可以根据应用程序需求稍后进行调整。

Figure 29. Project manager settings



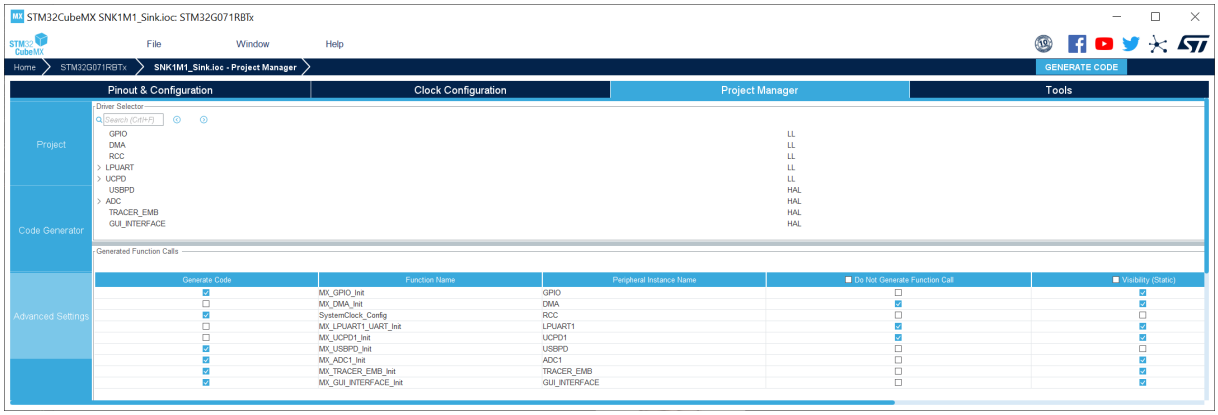
STMicroelectronics建议在“代码生成器”选项卡中，选中“添加必要的库文件作为参考”选项卡。

Figure 30. **Code generator settings**



Click on *Advanced Settings*
LPUART is selected as LL to save a bit of memory heap size.

Figure 31. **Project advanced setting**



Work must be saved: menu file / save

5.4 代码生成

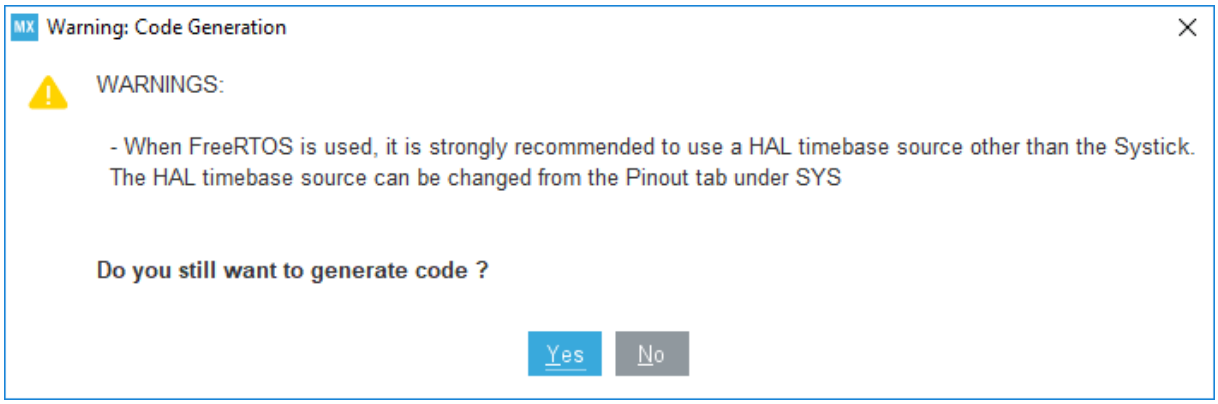
Click on generate code.

A warning appears, informing that a proper HAL timebase is not defined.

It is safer to use a dedicated timer as a HAL timebase source.

Note: his becomes the recommended standard way of working in the forthcoming firmware package deliveries, especially when usingSIS OS V, which defines Systick as FreeROS™ timebase. For this demonstration, the below warning can be ignored by clicking Yes.

Figure 32. Generation warning

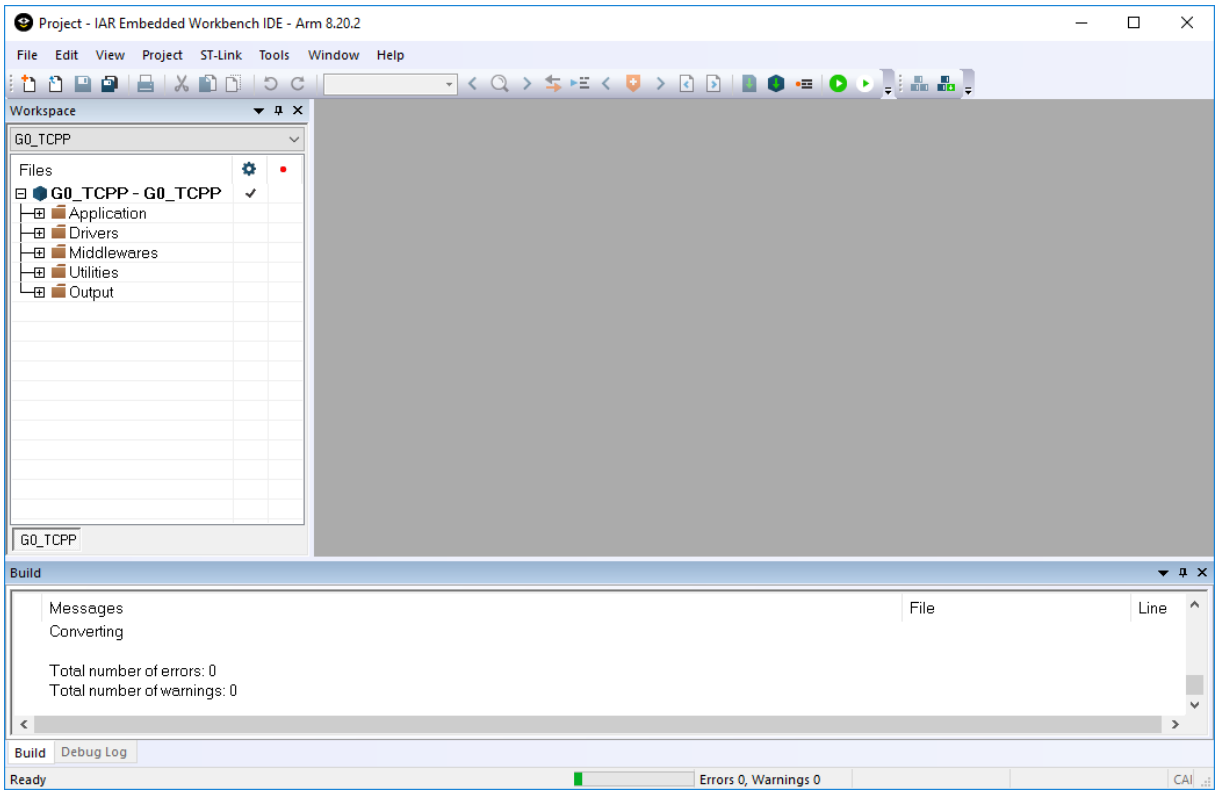


Then it is recommended to initialize Git to experiment with some code and be able to roll back to previous versions, as in classic software development.

5.5 编译生成的应用程序

The compilation must be performed without error or warning.

Figure 33. First compilation



In this project, different folders can be found:

- The `Application/User` folder contains the source files that we need to edit to enrich the application.
- The `Drivers` folder contains the HAL drivers for the STM32.
- The `Middleware` folder contains the source files and the libraries for FreeRTOS™ and USB-PD.
- The `Utilities` folder contains the GUI (UCPD monitor) and tracer embedded source files part.
- The `Output` folder contains the compilation result files.

5.6 完整的USB-PD应用

既然外围设备已由STM32CubeMX初始化，则需要添加一些最低级别的应用程序：

- ADC needs to be calibrated, and conversion needs to start.
- Fill the handlers for the interrupts to wake up the UCPD peripheral.
- Fill *BSP_USBPD_PWR_VBUSGetVoltage* function with the right coefficient depending on the V_{BUS} divider bridge.
- Complete *USBPD_DPM_SNK_EvaluateCapabilities* to answer one source capability message.
- TCPP01-M12 dead battery pin needs to be disabled, GPIO driven HIGH, to see the source Rp, or the jumper has to be set on the shield.

5.6.1 Modification in main.c

In this file, the ADC must start after its calibration, using HAL. The ADC is needed to read V_{BUS}.

Code to be added between `USER CODE ADC1_Init 2` tags:

```
…
 /* USER CODE BEGIN ADC1_Init 2 */
   HAL_ADCEx_Calibration_Start(&hadc1);
  HAL_ADC_Start(&hadc1);
  /* USER CODE END ADC1_Init 2 */
…
```

Note: For STM32G4, ADC calibration API is different, the calibration line must be replaced by:

```
HAL_ADCEx_Calibration_Start(&hadc1, sConfig.SingleDiff);
```

Note: This simple example is not optimized from a power point of view, as the ADC is always running.

5.6.2 Modification in usbpd_dpm_user.c

To avoid a hard fault if the distant device asks for sink capabilities, some code must be added inside the *USBPD_DPM_GetDataInfofunction* function.

In the case, before the default add :

```
case USBPD_CORE_DATATYPE_SNK_PDO: /*!< Handling of port Sink PDO, requested by get sink
capa*/
  USBPD_PWR_IF_GetPortPDOs(PortNum, DataId, Ptr, Size);
  *Size *= 4;
  break;
```

The *USBPD_DPM_SNK_EvaluateCapabilities* function needs to be added to establish the first contract. It is a very basic example that requests the first default 5V PDO. This must be modified to match with real SINK PDOs, which are not yet managed by STM32CubeMX.

In the user code for *USBPD_DPM_SNK_EvaluateCapabilities* replace

```
DPM_USER_DEBUG_TRACE(PortNum, "ADVICE: update USBPD_DPM_SNK_EvaluateCapabilities");
```

with the following text:

```
…
/* USER CODE BEGIN USBPD_DPM_SNK_EvaluateCapabilities */
 USBPD_SNKRDO_TypeDef rdo;
  /* Initialize RDO */
  rdo.d32 = 0;
  /* Prepare the requested pdo */
  rdo.FixedVariableRDO.ObjectPosition = 1;
  rdo.FixedVariableRDO.OperatingCurrentIn10mAunits  = 50;
  rdo.FixedVariableRDO.MaxOperatingCurrent10mAunits = 50;
  rdo.FixedVariableRDO.CapabilityMismatch = 0;

  *PtrPowerObjectType = USBPD_CORE_PDO_TYPE_FIXED;
  *PtrRequestData = rdo.d32;
/* USER CODE END USBPD_DPM_SNK_EvaluateCapabilities */
…
```

Note: `ADVICE` keyword is used to indicate to the user that he may need to add his code to get a functional application.

5.6.3 Modification in usbpd_pwr_user.c

It is important to add this part to correctly read V_{BUS} provided by the ADC. The stack needs to know the V_{BUS} level all along the cable presence to determine the action to take. In the case of SINK, the detachment is done when V_{BUS} is below vSafe0V.

```
…
/* USER CODE BEGIN include */
#include "main.h"
/* USER CODE END include */
…
/* USER CODE BEGIN BSP_USBPD_PWR_VBUSGetVoltage */

  /* Check if instance is valid       */
  int32_t ret = BSP_ERROR_NONE;

  if ((Instance >= USBPD_PWR_INSTANCES_NBR) || (NULL == pVoltage))
  {
    ret = BSP_ERROR_WRONG_PARAM;
    *pVoltage = 0;
  }
  else
  {
    uint32_t val;
    val = __LL_ADC_CALC_DATA_TO_VOLTAGE( VDDA_APPLI, \
      LL_ADC_REG_ReadConversionData12(ADC1), \
        LL_ADC_RESOLUTION_12B); /* mV */
    /* X-NUCLEO-USBPDM board is used */
    /* Value is multiplied by 5.97 (Divider R6/R7 (40.2K/200K) for VSENSE) */
    val *= 597;
    val /= 100;
    *pVoltage = val;
  }
  return BSP_ERROR_NONE;

/* USER CODE END BSP_USBPD_PWR_VBUSGetVoltage */
…
```

The calculation of the `val` variable depends on the voltage divider shown in Figure 14. On the X-NUCLEO-USBPDM1 shield, `Value` is multiplied by 5.97 (Divider R6/R7 40.2 kΩ/200 kΩ) for VSENSE.

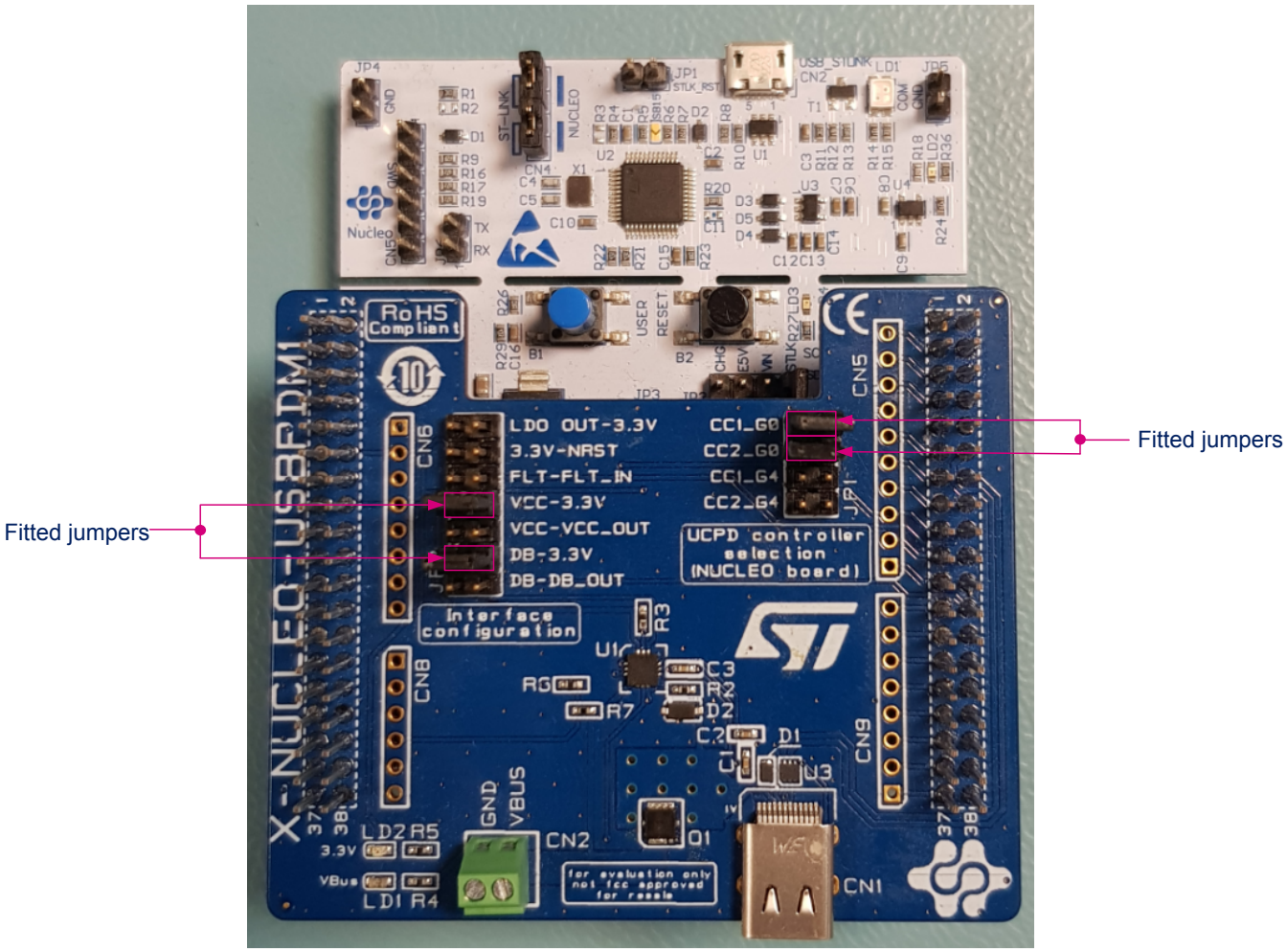
Note: In the X-CUBE-TCPP project, for the `Projects\NUCLEO-G071RB\Applications\USB_PD\USBPDM1_Sink_PPS` application, the `.extsettings` file is used to exercise the BSP shield (available in `Drivers/BSP/X-NUCLEO-USBPDM1` directory). Doing so, the weak functions in the generated code for the power parts are overloaded by the BSP files, and there is no need to manually modify the files.

5.7 检查跳线

This is the last jumper setting check before the first power delivery contract.

5.7.1 X-NUCLEO-USBPDM1 jumpers

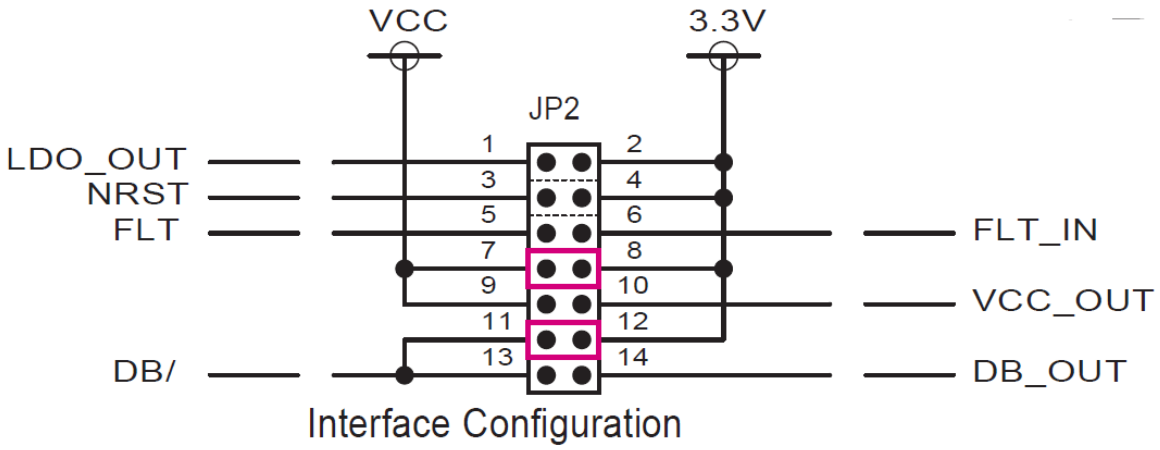
Figure 34. **X-NUCLEO-USBPDM1 shield picture**



Verify that the two JP1 jumpers located on the right to select the STM32G0 and STM32G4 configuration are inserted.

Then select the pins that are controlled by the MCU, using the left JP2 jumpers:

Figure 35. TCPP01-M12 jumper settings for X-NUCLEO-USBPDM1

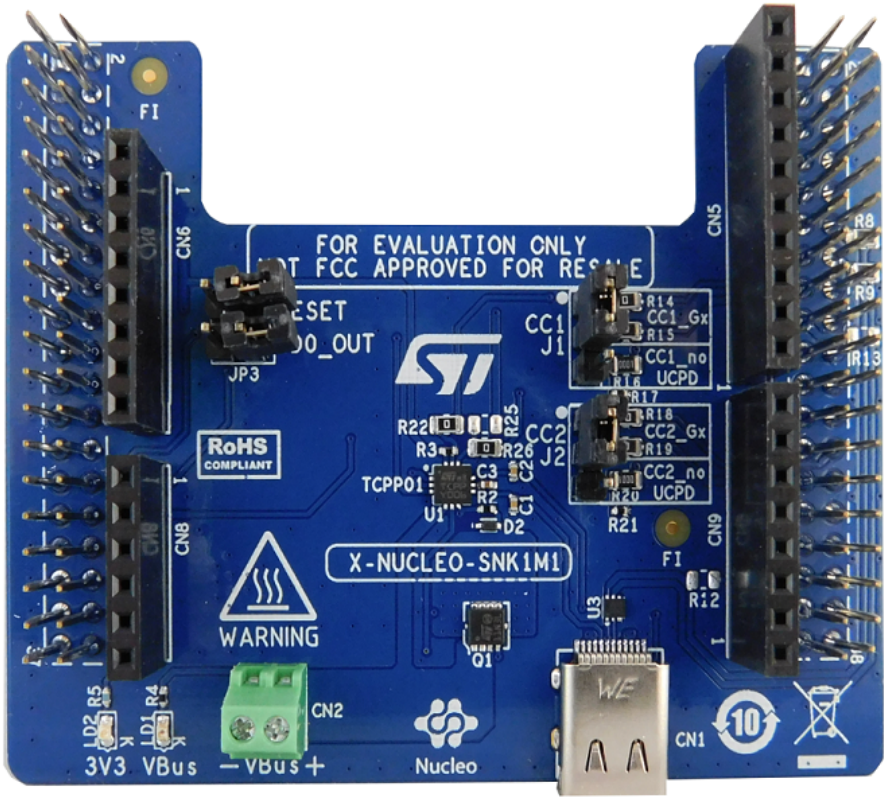


- Fault detection and hard reset are not managed in this demonstration.
- The power consumption is also not optimized. This is the reason why the TCPP01-M12 VCC is set to the fixed 3.3 V, instead of taking an MCU GPIO, so the JP2 jumper VCC-3.3V [7-8] is ON
- In the first step demonstration, the dead battery from the TCPP01-M12 is not used, so the JP2 jumper DB-3.3V [11-12] is ON.

5.7.2 X-NUCLEO-SNK1M1 jumpers

The JP3 jumpers are needed to select the STM32 power supply, from V_{BUS} or ST-LINK. For now, both jumpers need to be left open to allow download and debugging from ST-LINK. Refer to Figure 36.

Figure 36. X-NUCLEO-SNK1M1 top view

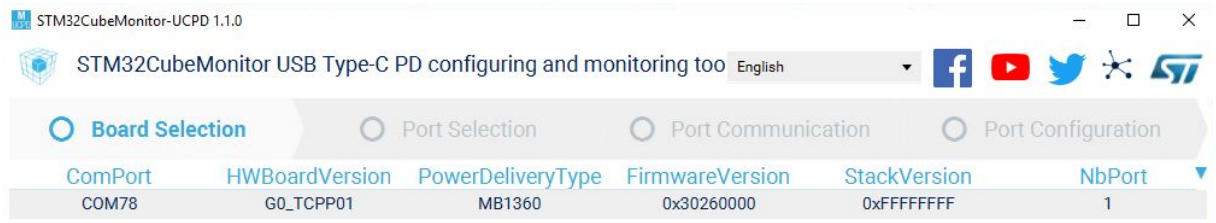


6 建立第一个明确的合同

编译应用程序，刷新开发板，启动STM32G0程序，将USB电缆插入，因为必须使用Virtual COM端口，然后启动UCPD监视器。

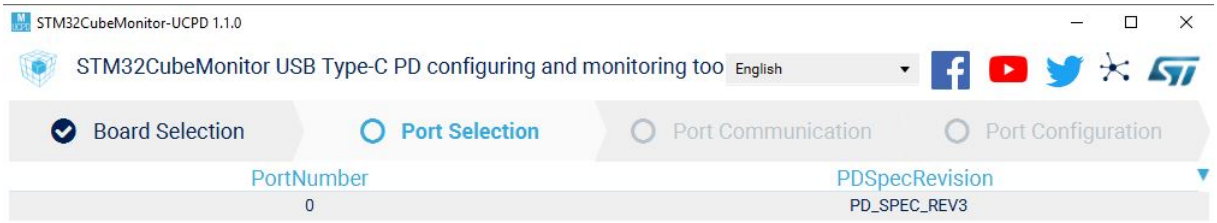
单击“刷新已连接板的列表”时，用户板必须出现在列表中，因此双击相应的行（或单击“下一步”）。

Figure 37. Board selection



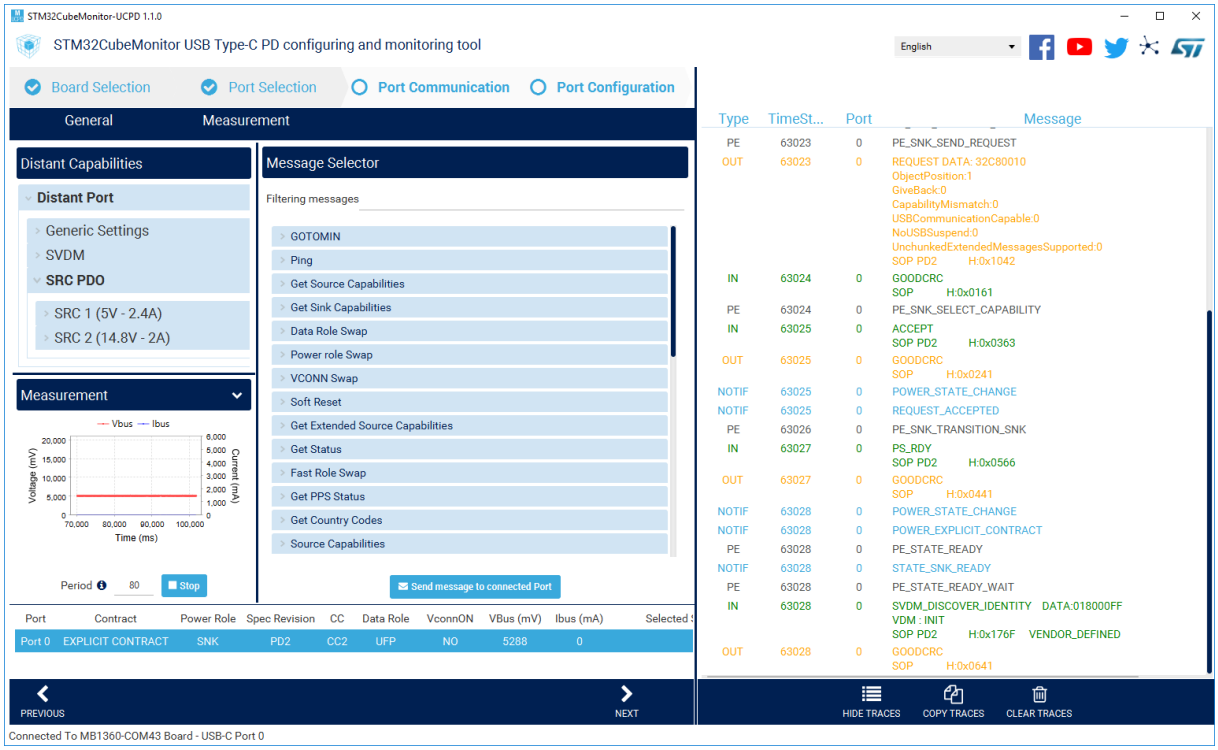
Note: *T*ComPort可能有所不同。 这取决于计算机上安装的板的数量。 然后双击所需的UCPD端口（此处为端口0），或选择它并单击“下一步”。

Figure 38. Port selection



Click on the "TRACES" button in the bottom right corner to get protocol traces. Then it is possible to plug a power delivery source cable into the USB Type-C® receptacle of the X-NUCLEO-USBPDM1 shield. The screen may look like Figure 39:

Figure 39. Explicit contract visible in UCPD monitor



Note: The SRC PDO part may look different. It depends on the capabilities of the power source.

Figure 39 shows the communication between the STM32G0 and the power delivery source on the right panel. It is possible to verify the correct sequence to reach an explicit contract:

1. The capabilities are sent by the source (IN green message).
2. The request is sent by the STM32G0 (OUT orange message).
3. The ACCEPT and the PS_RDY are sent by the source (IN green message).

For more details on how to use this tool, refer to UM2468.

And for more details on the protocol, refer to UM2552.

Note that this trace is very helpful for debugging and application development.

6.1 How to debug a bit deeper

6.1.1 livewatch variable setting

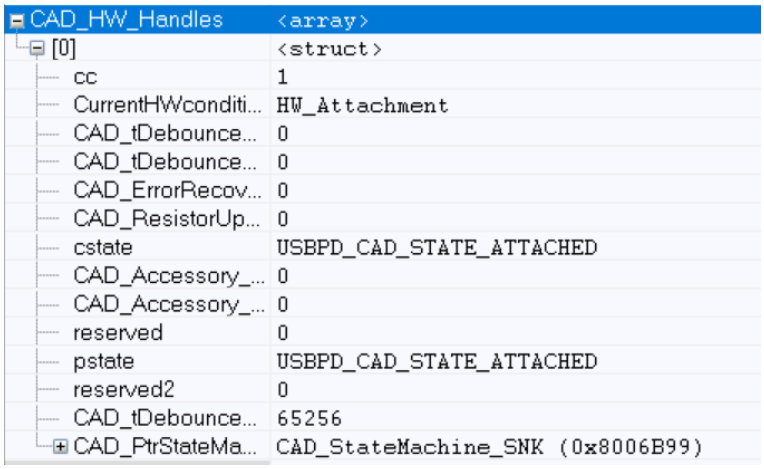
For more information in trace about CAD state machine, refer to Figure 40, where CAD appears in column *Type*.

Figure 40. Example of CAD debug information visible in the trace

Type	TimeSt...	Port	Message
CAD	906209	0	USBPD_CAD_STATE_SWITCH_TO_SRC
EVENT	906209	0	EVENT_DETACHED
NOTIF	906209	0	POWER_STATE_CHANGE
DEBUG	906209	0	HELP: update USBPD_DPM_SetDataInfo:7
DEBUG	906209	0	HELP: update USBPD_DPM_SetDataInfo:2
PE	906209	0	PE_SNK_STARTUP
CAD	906209	0	USBPD_CAD_STATE_DETACHED
DEBUG	914101	0	HELP: Update BSP_PWR_VBUSInit
CAD	914101	0	USBPD_CAD_STATE_ATTACHED_WAIT
CAD	914316	0	USBPD_CAD_STATE_ATTACHED0

Add *livewatch* on CAD_HW_Handles. This variable can be used to check the Type-C attachment or detachment.

Figure 41. cstate=1: detached



If the CC lines level changes are invisible, check that the TCPP is powered on and the active low _DB pin is not set at 3.3 V. It may come from the JP2 jumper or some GPIO settings, like pull-up resistors.

Note: In the current STM32CubeMX for STM32G4, there is an issue with the default GPIO mode for CC2. In usbpd_cad_hw_if.c there must be:

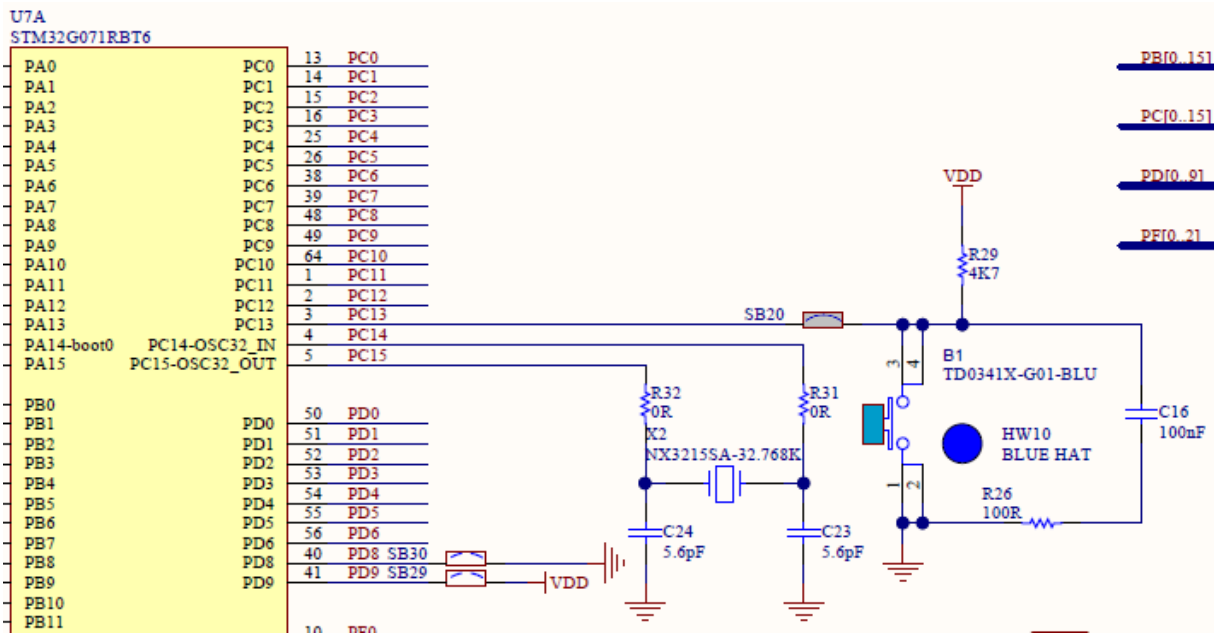
```
LL_GPIO_SetPinMode(GPIOB, LL_GPIO_PIN_4, LL_GPIO_MODE_ANALOG);
```

In STM32G4 versions before firmware 1.2.0, the correct compilation switch is not set. An easy way to correct this issue is to activate the compilation switch MB1367.

6.1.2 User button

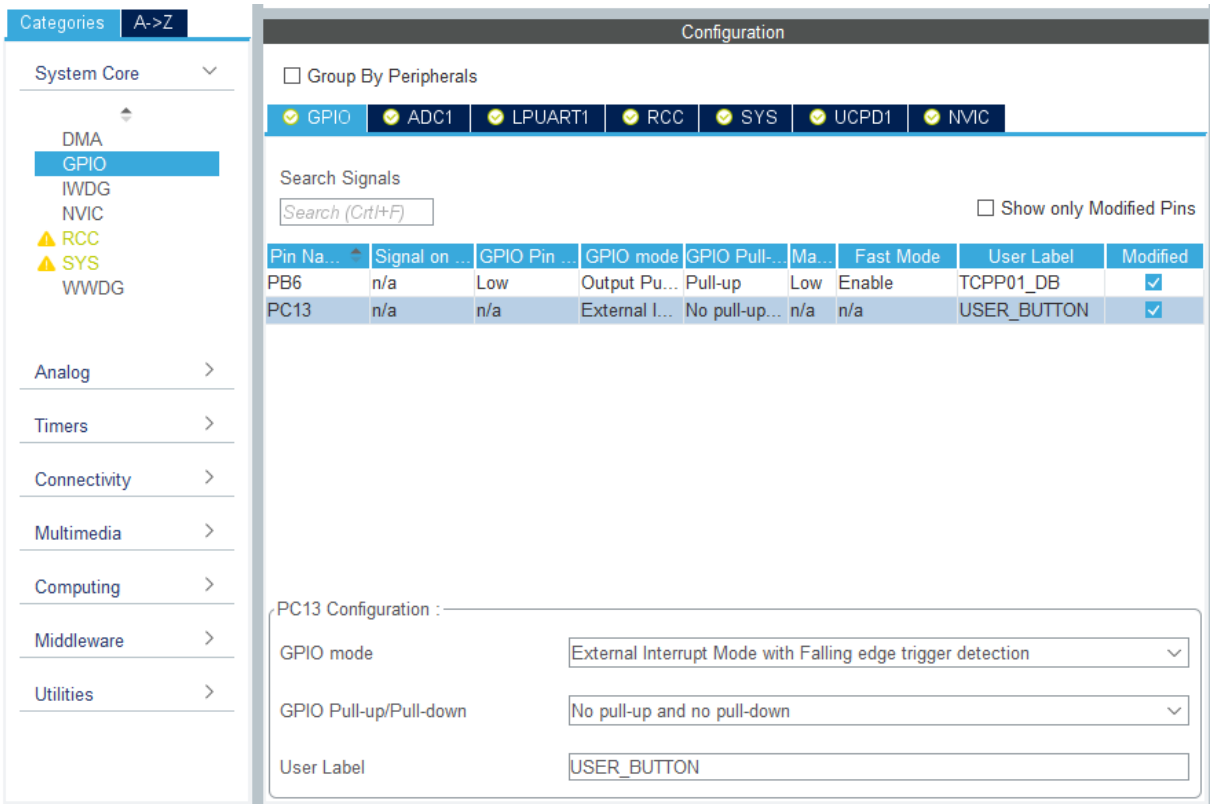
For further debugging, the V_{BUS} measured value can be printed in the trace, using the user button:

Figure 42. User button on STM32G0 Nucleo-64 board schematics



Add the button from STM32CubeMX as described in Figure 43.

Figure 43. Add the user button in STM32CubeMX



Add in src/main.c:

```
/**
  * @brief EXTI line detection callbacks
  * @param GPIO_Pin Specifies the pins connected EXTI line
  * @retval None
  */
void HAL_GPIO_EXTI_Falling_Callback(uint16_t GPIO_Pin)
{
  if (GPIO_Pin == USER_BUTTON_PIN) /* Will display in trace the VBUS value when user button
is pressed */
  {
    char _str[10];
    BSP_PWR_VBUSGetVoltage(0);
    sprintf(_str,"VBUS:%d", BSP_PWR_VBUSGetVoltage(0));
    USBPD_TRACE_Add(USBPD_TRACE_DEBUG, 0, 0, (uint8_t*)_str, strlen(_str));
  }
}
```

And the corresponding interrupt in src/stm32g0xx_it.c:

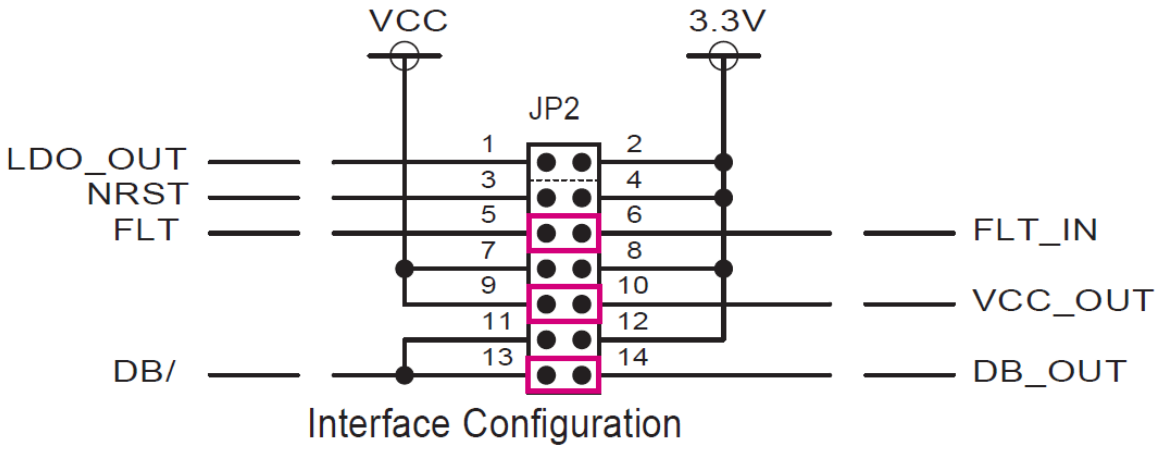
```
/**
  * @brief  This function handles the external line 4_15 interrupt request.
  * @retval None
  */
void EXTI4_15_IRQHandler(void)
{
  HAL_GPIO_EXTI_IRQHandler(USER_BUTTON_PIN);
}
```

7 Next steps

X-NUCLEO-USBPDM1 can be better controlled by driving its dead battery pin and its power like it is described in Section 5.1.6 for X-NUCLEO-SNK1M1.

If these pins need to be controlled by the application, in case the X-NUCLEO-USBPDM1 shield is used, the jumper positions must be adapted. The potential faults must also be read by setting the JP2 jumper in [5-6] position.

Figure 44. X-NUCLEO-USBPDM1 shield jumpers position when application manages FLT, DB, and the TCPP01-M12 VCC



The user application also needs to be done to react upon the TCPP01-M12 fault detection (Over-temperature, over-voltage).

8 Conclusion

This demonstration is only the first step to a power delivery application.

This quickly developed application is not optimized from the low-power point of view.

The USB PD application performed here is the bare minimum. No software code is added to select the power level by looking at the proposal sent by the source. The mandatory hard reset management is also missing.

To continue further, various demonstrations are available on STM32G0, STM32G4, and STM32L5. Check the Projects directory in the firmware package available in each serie, on the ST website. For instance, UCPD demonstration on EVAL_G0 is available under the folder `.\Projects\STM32G081B-EVAL\Demonstrations\DemoUCPD`.

Revision history

Table 2. **Document revision history**

Date	Version	Changes
9-Jan-2020	1	Initial release
30-Apr-2020	2	Added specific technical data throughout the document
27-Nov-2020	3	Updated: Figures: • *Figure 7. FreeRTOS™ configuration* • *Figure 11. Detailed stack configuration* • *Figure 24. Selection of USB-PD middleware TRACER_EMB source* • *Figure 26. Project manager settings* • *Figure 34. Port selection* Code blocks in: • *Section 5.6.2 Modification in stm32g0xx_it.c* • *Section 5.6.3 Modification in usbpd_dpm_user.c* • *Section 5.6.4 Modification in usbpd_pwr_user.c*
3-May-2021	4	Added: • X-NUCLEO-SNK1M1 shield • Links in Introduction • Figure 2. STM32G0 Nucleo-64 board equipped with X-NUCLEO-SNK1M1 shield • Figure 3. X-CUBE-TCPP block diagram architecture • Figure 20. Modify TCPP01-M12 controls • TCPP in Acronym definitions • Section 5.1.6 Additional GPIO settings • Section 5.7.2 X-NUCLEO-SNK1M1 jumpers Updated: • UM2668 replaced by UM2773 and YouTube video link in Section 3 Reference documents • Figure 14. TCPP01-M12 shield voltage divider • Figure 15. STM32G0 Nucleo-64 board (left) and TCPP01-M12 shield (right) schematics • Figure 31. Project advanced setting • `usbpd_cad_hw_if.c` code and Figure 41 in Section 6.1.1 livewatch variable setting • Section 7 Removed: • Former 5.6.2 section on *Modification in stm32g0xx_it.c*

Contents

List of tables

List of figures