ModusToolbox™软件中的 EZ-PD™ PMG1 MCU 入门指南

关于本文档

范围和目的

本应用笔记向您介绍在具有 PD 功能的高电压微控制器系列中 EZ-PD™ PMG1 (第一代 PD 微控制器) 系列的功能，并帮助您在 ModusToolbox™软件的 Eclipse IDE 中通过使用 PMG1 MCU 来开始您的第一个项目。

目标读者

本文档的主要目标读者是使用 ModusToolbox™软件开发 EZ-PD™ PMG1 MCU 的所有对象。

Table of contents

关于本文档 1
Table of contents 1
1 简介 3
1.1 PMG1 MCU 系列概述和比较 3
1.2 PMG1-S0 MCU 5
1.3 PMG1-S1 MCU 6
1.4 PMG1-S2 MCU 7
1.5 PMG1-S3 MCU 8
1.6 软件环境：ModusToolbox™软件 9
2 开发 10
2.1 PMG1 MCU 资源 10
2.2 使用 ModusToolbox™软件开发固件/应用程序 10
2.2.1 PMG1 MCU 的软件资源 11
2.2.1.1 配置 12
2.2.1.2 PMG1 MCU 的软件开发 12
2.3 ModusToolbox™软件帮助 13
2.4 其它 IDE 的支持 13
2.5 编程与调试 13
2.6 PMG1 MCU 套件 14
3 我的第一个基于 Eclipse IDE 的 Modus Toolbox™软件设计 15
3.1 先决条件 15
3.1.1 硬件 15
3.1.2 软件 15
3.2 应用程序开发流程 15
3.3 设计概述 15
3.4 实现流程 16
3.4.1 第一部分：创建新的应用 16
3.4.2 第二部分：查看和修改设计 18
3.4.3 第三部分：编写固件 20
3.4.3.1 固件流程 20

Table of contents

3.4.4 第四部分：构建应用程序 22
3.4.5 第五部分：编程器件 24
3.4.6 第六部分：测试您的设计 26
3.4.7 第七部分：调试应用程序 27
3.4.7.1 通过 KitProg3 接口进行调试 27
3.4.7.2 通过 10 引脚 SWD 插头进行调试 30
3.5 添加中介软件 31
3.6 USB-PD 接收源示例 31
3.7 在各个 PMG1 MCU 之间导入项目 32
4 总结 35
相关文档 36
缩略语与首字母缩略词 37
术语表 38
修订记录 39

1 简介

1.1 PMG1 MCU 系列概述和比较

EZ-PD™ PMG1 (第一代 PD 微控制器) 是具有电力传输 (PD) 应用的高电压微控制器成员之一。这些芯片拥有一个 Arm® Cortex® -M0/M0+ CPU、USB PD 控制器以及模拟和数字外设。PMG1 MCU 适用于嵌入式系统，这些系统为高电压 USB PD 端口提供电源或消耗来自该端口的电源，并利用微控制器提供额外的控制功能。

Table 1 显示了 PMG1 MCU 系列中不同 MCU 的性能对比。您可以通过该表为您的应用选择合适的 PMG1 MCU。

Table 1 PMG1 MCU 系列中不同 MCU 的性能对比

子系统或范围	条目	PMG1-S0	PMG1-S1	PMG1-S2	PMG1-S3
CPU 和存储器子系统	内核	Arm® Cortex®-M0	Arm® Cortex®-M0	Arm® Cortex®-M0	Arm® Cortex®-M0+
	最大频率 (MHz)	48	48	48	48
	闪存 (KB)	64	128	128	256
	SRAM (KB)	8	12	8	32
电力传输	电力传输端口	1	1	1	48-QFN 具有一个端口 97-BGA 具有两个端口
	功能	接收源	DRP	DRP	DRP
	MOSFET 门控驱动程序	1x PFET	2x PFET	2x NFET	灵活的 2x NFET
	故障保护	VBUS OVP 和 UVP	VBUS OVP, UVP 和 OCP。SCP 和 RCP (仅限发送源)	VBUS OVP, UVP 和 OCP	VBUS OVP, UVP 和 OCP。SCP 和 RCP (仅限发送源)
USB	支持公告牌类的集成全速 USB 2.0 器件	不支持	不支持	支持	支持
电压范围	电源 (V)	VDDD (2.7–5.5) VBUS (4–21.5)	VSYS (2.75–5.5) VBUS (4–21.5)	VSYS (2.7–5.5) VBUS (4–21.5)	VSYS (2.8–5.5) VBUS (4–28)
	IO (V)	1.71–5.5	1.71–5.5	1.71–5.5	1.71–5.5
数字模块	SCB (可配置为 I2C/UART/SPI)	2	4	4	48-QFN 具有七个 (其中只有五个可以配置为 SPI 和 UART) 97-BGA 具有八个
	TCPWM 模块 (可配置为定时器, 计数器或脉宽调制器)	4	2	4	48-QFN 具有七个, 97-BGA 具有八个

简介

子系统或范围	条目	PMG1-S0	PMG1-S1	PMG1-S2	PMG1-S3
	硬件身份认证模块 (加密)	不支持	不支持	支持 (AES-128/192/256, SHA1, SHA2-224, SHA2-256, PRNG, CRC)	支持 (AES-128, SHA2-256, TRNG, 向量单位)
模拟模块	模数转换器 (ADC)	2x 8 位 SAR	1x 8 位 SAR	2x 8 位 SAR	2x 8 位 SAR 1 x 12 位 SAR
	片上温度感应器	支持	支持	支持	支持
直接储存器访问 (DMA)	DMA	不支持	不支持	不支持	支持
GPIO	最大输入/输出数量	12 个 (10 个+2 个 OVT)	17 个 (15 个+2 个 OVT)	20 个 (18 个+2 个 OVT)	48-QFN 具有 26 个 (24 个+2 个 OVT) 97-BGA 具有 50 个 (48 个+2 个 OVT)
充电标准	充电发送源	-	BC 1.2, AC	BC 1.2, AC	BC 1.2, AC, AFC 以及快速充电 3.0
	充电接收源	BC 1.2, 苹果充电 (AC)	BC 1.2, AC	BC 1.2, AC	BC 1.2, AC
ESD 保护	ESD 保护	支持 (接触放电可达 ± 8 kV, 空气放电可达 ±15 kV, HBM, CDM)	支持 (HBM 和 CDM)	支持 (接触放电可达 ± 8 kV, 空气放电可达 ±15 kV, HBM, CDM)	支持 (HBM 和 CDM)
封装类型	封装选项	24-QFN (4x4 mm, 间隙为 0.5 mm)	40-QFN (6×6 mm, 间隙为 0.5 mm)	40-QFN (6×6 mm, 间隙为 0.5 mm)	48-QFN (6x6 mm, 间隙为 0.5 mm) 97-BGA (6x6 mm, 间隙为 0.5 mm 和 0.65 mm)

1.2 PMG1-S0 MCU

PMG1-S0 MCU 是 PMG1 MCU 系列的第一个成员。它包括一个具有 64 KB 闪存的 48 MHz Arm® Cortex®-M0 处理器、一个完整的 Type-C USB-PD 收发器、用于支持 Type-C 端口上的接收源的下拉终端电阻 R_D、12 个 GPIO 和系统级 ESD 保护。该芯片采用了 24-QFN 的封装形式。

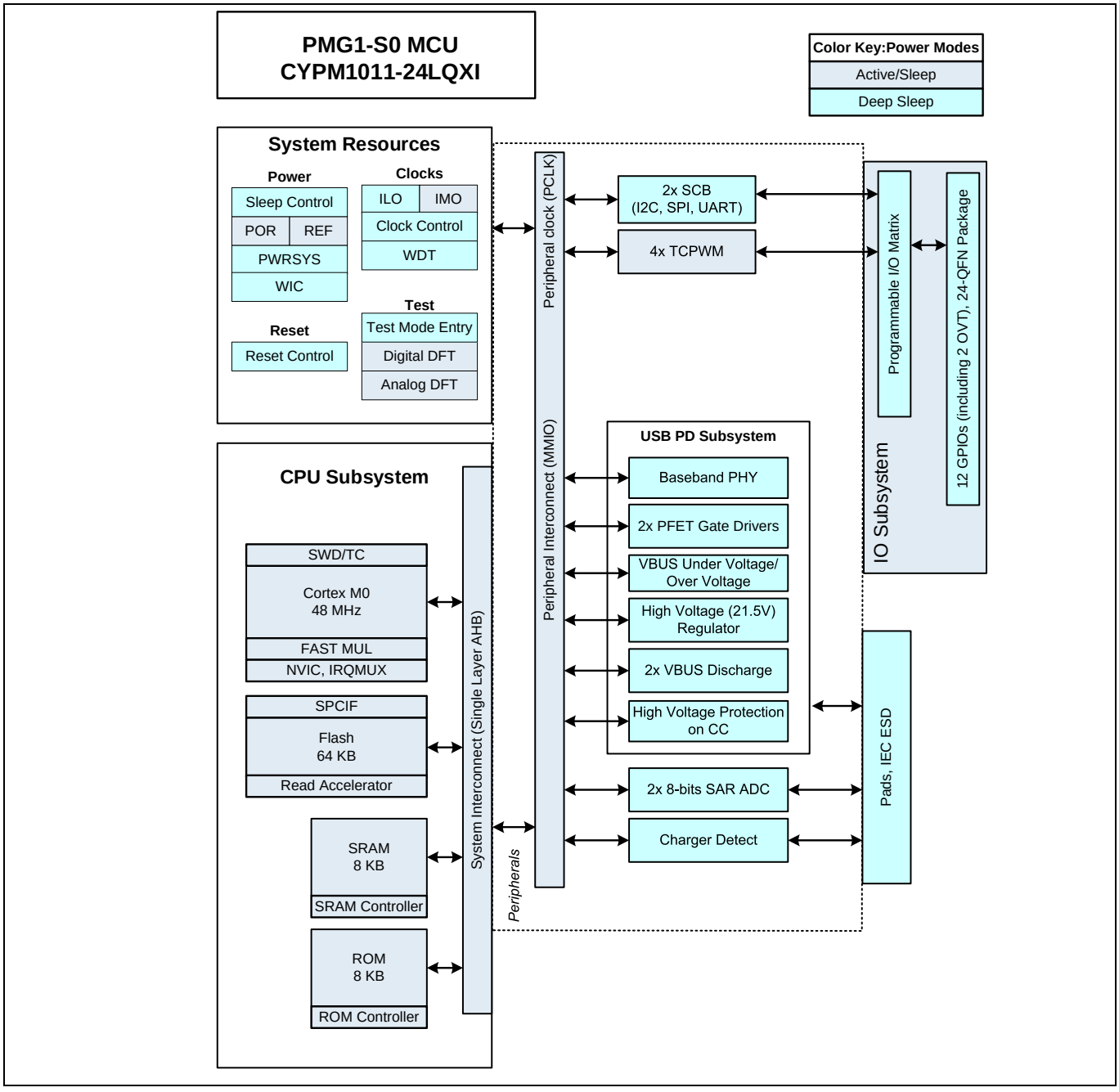


Figure 1 PMG1-S0 MCU 框图

1.3 PMG1-S1 MCU

PMG1-S1 MCU 包括一个 48 MHz Arm® Cortex®-M0 处理器，但它的闪存容量比 PMG1-S0 MCU 的闪存容量大。该 MCU 具有 128 KB 闪存、一个带有所有终端电阻 R_P 和 R_D 的完整 Type-C USB PD 收发器、亏电电池 R_D，以及 17 个 GPIO。该器件采用了 40-QFN 的封装。

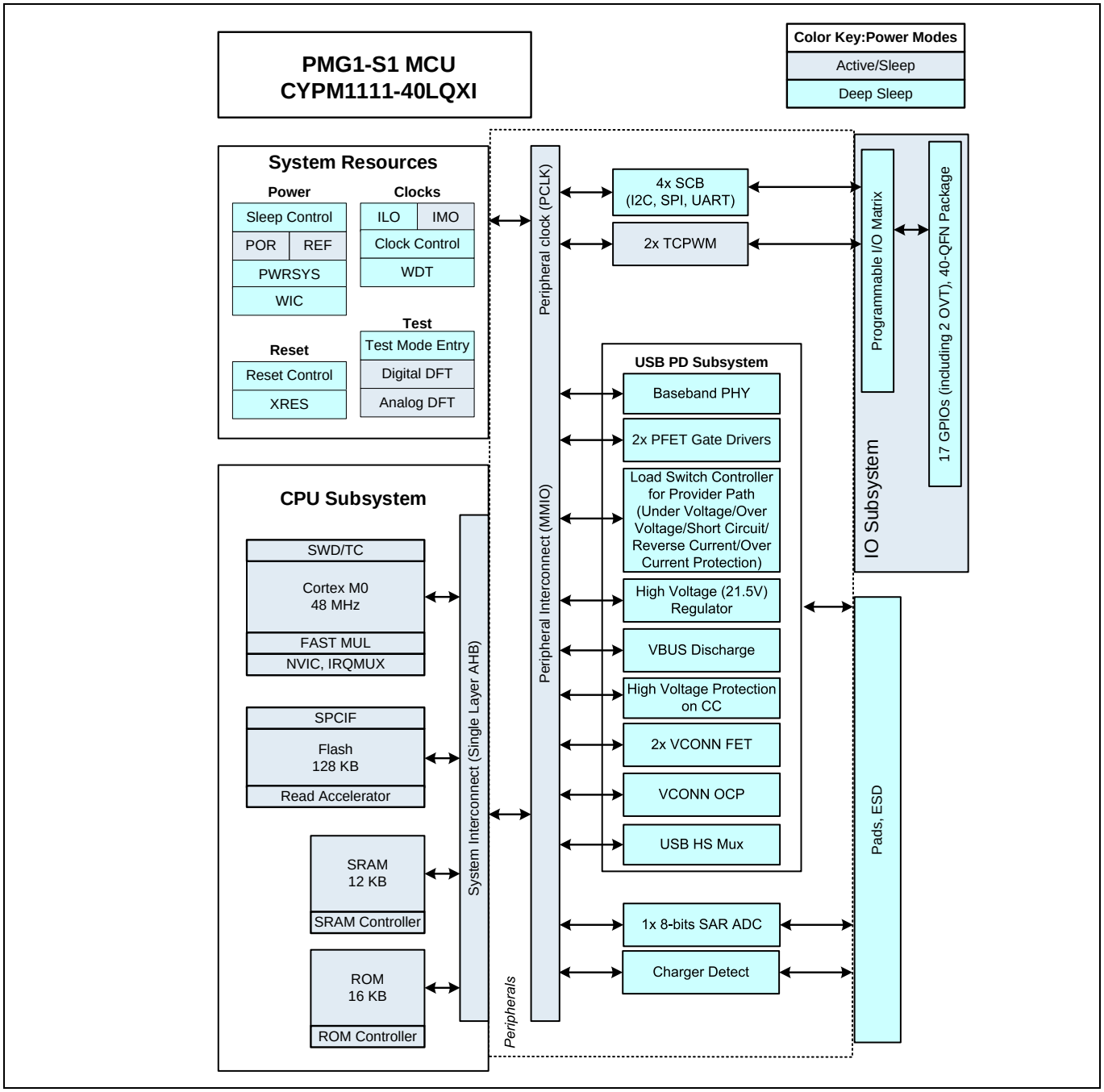


Figure 2 PMG1-S1 MCU 框图

1.4 PMG1-S2 MCU

PMG1-S2 MCU 使用了一个带有 128 KB 闪存的 48 MHz Arm® Cortex®-M0 处理器、一个全速 USB 器件控制器、一个用于身份认证的加密引擎、20 个 GPIO 和系统级 ESD 保护。PMG1-S2 MCU 的封装形式为 40-QFN。

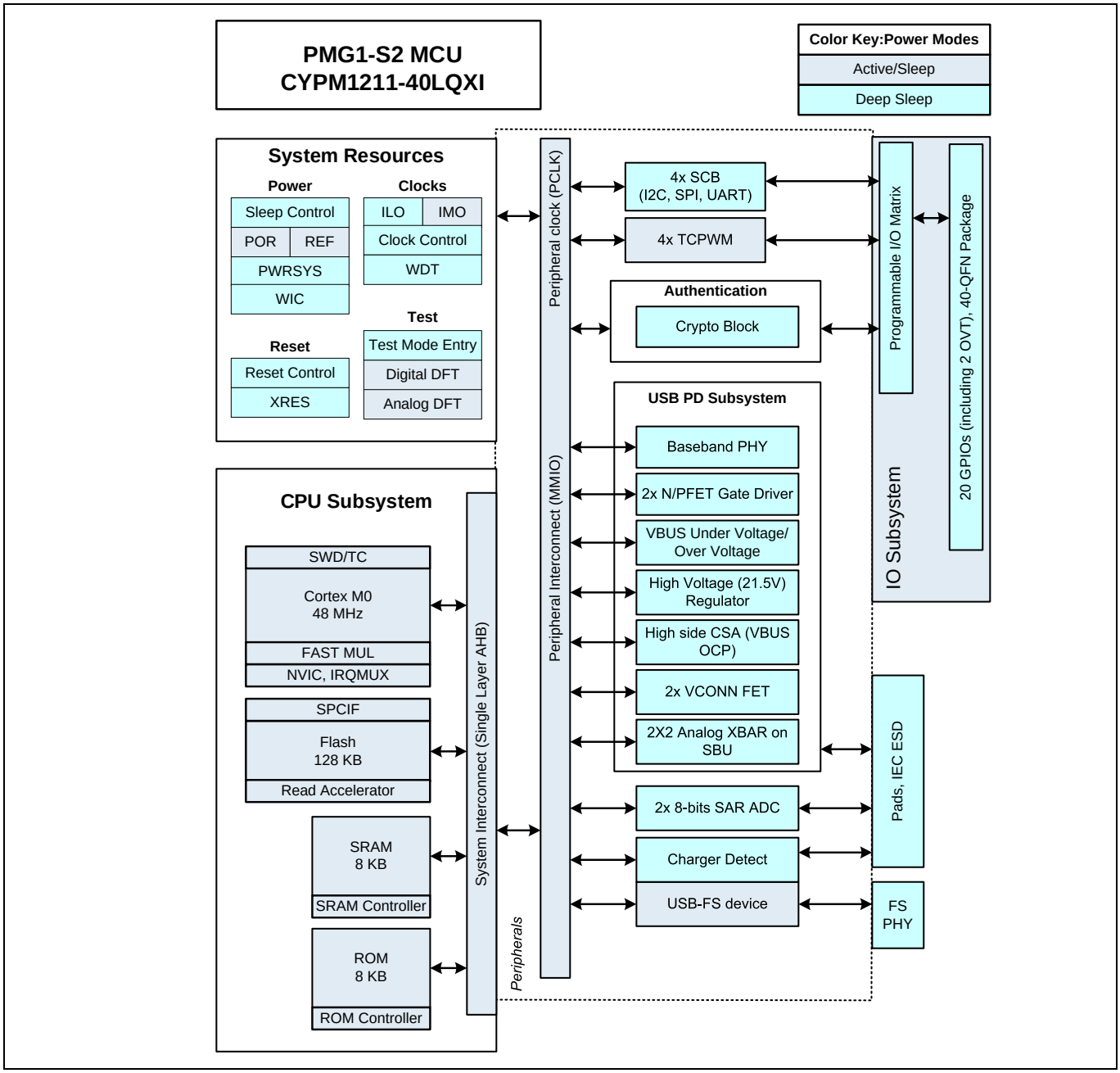


Figure 3 PMG1-S2 MCU 框图

1.5 PMG1-S3 MCU

PMG1-S3 MCU 使用一个带有 256 KB 闪存的 48 MHz Arm® Cortex®-M0+处理器、一个全速 USB 器件控制器、CAPSENSE™、DMA、一个用于身份认证的加密引擎、26/50 个 GPIO 和系统级 ESD 保护。PMG1-S3 MCU 的封装形式为 48-QFN 和 97-BGA。

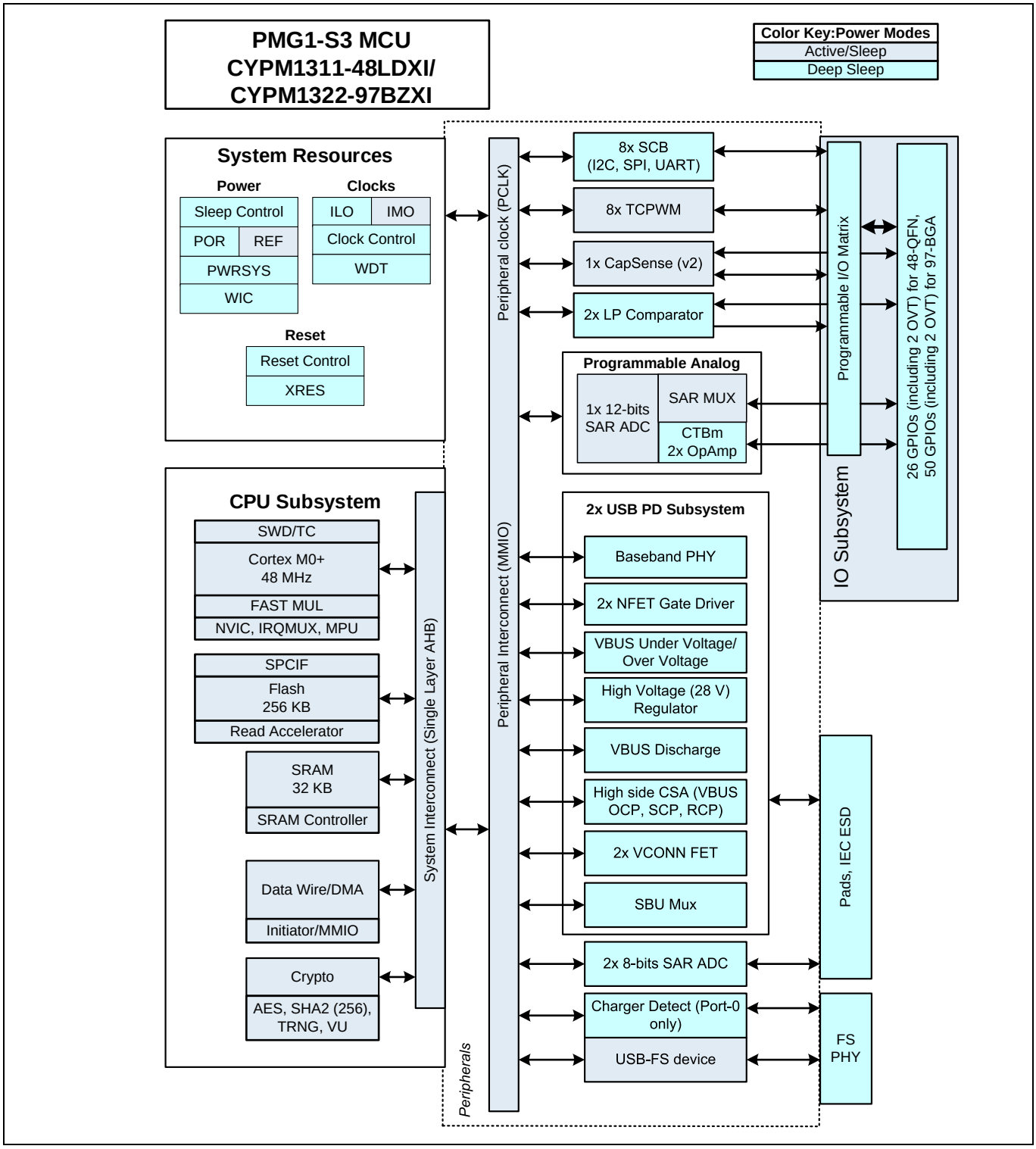


Figure 4 PMG1-S3 MCU 框图

1.6 软件环境：ModusToolbox™软件

ModusToolbox®软件环境通过一组工具支持 PMG1 MCU 应用程序开发，可通过这些工具来配置器件、设置外设以及使用世界一流的中介软件补充您的项目。

本应用笔记概述了 ModusToolbox™开发生态系统，并帮助您从一个简单的“Hello World” (你好世界) 应用程序入手。下面的内容介绍了从空模板应用程序创建一个应用程序的详细流程。

2 开发

2.1 PMG1 MCU 资源

PMG1 MCU 系列具有一系列丰富的文档、开发工具和在线资源，能够在开发过程中为您提供帮助。下面提供了 PMG1 MCU 的简要资源列表：更多相关信息，请访问 **PMG1 MCU 网页**。

- PMG1 MCU 网页
- 数据手册提供了选择和使用特定器件的必要信息，其中包括功能说明和电气规范。
- 应用笔记和代码示例包含了从基本到高级的广泛主题。
- 技术参考手册 (TRM) 详细说明了每个器件系列中的架构和寄存器。
- 可使用原型开发套件来评估、设计和开发使用 PMG1 MCU 的不同应用。
- 技术支持：PMG1 MCU 社区论坛、知识库文章

2.2 使用 ModusToolbox™软件开发固件/应用程序

ModusToolbox™是用于开发使用 PMG1 MCU 的固件/应用程序的开发平台。它是最新一代的工具集，并具有 Eclipse IDE，因此 Windows、Linux 和 macOS 平台都支持该工具集。ModusToolbox™软件包含了配置器、低电平驱动程序、中间件库以及其它软件包，有助于您创建 PMG1 MCU 的应用程序。通过使用配置器，您可以设置器件中不同模块的配置，并生成可用于开发固件的代码。

ModusToolbox™的 Eclipse IDE 在快速面板中集成了工具和设计配置器的快速启动器。ModusToolbox™还支持第三方的 IDE，包括 Visual Studio Code、Arm® MDK (µVision) 和 IAR Embedded Workbench。

Figure 5 显示了 ModusToolbox™软件中包含的工具/资源的高级视图。有关 ModusToolbox™软件的更深入概述，请参阅 **ModusToolbox™用户指南**。

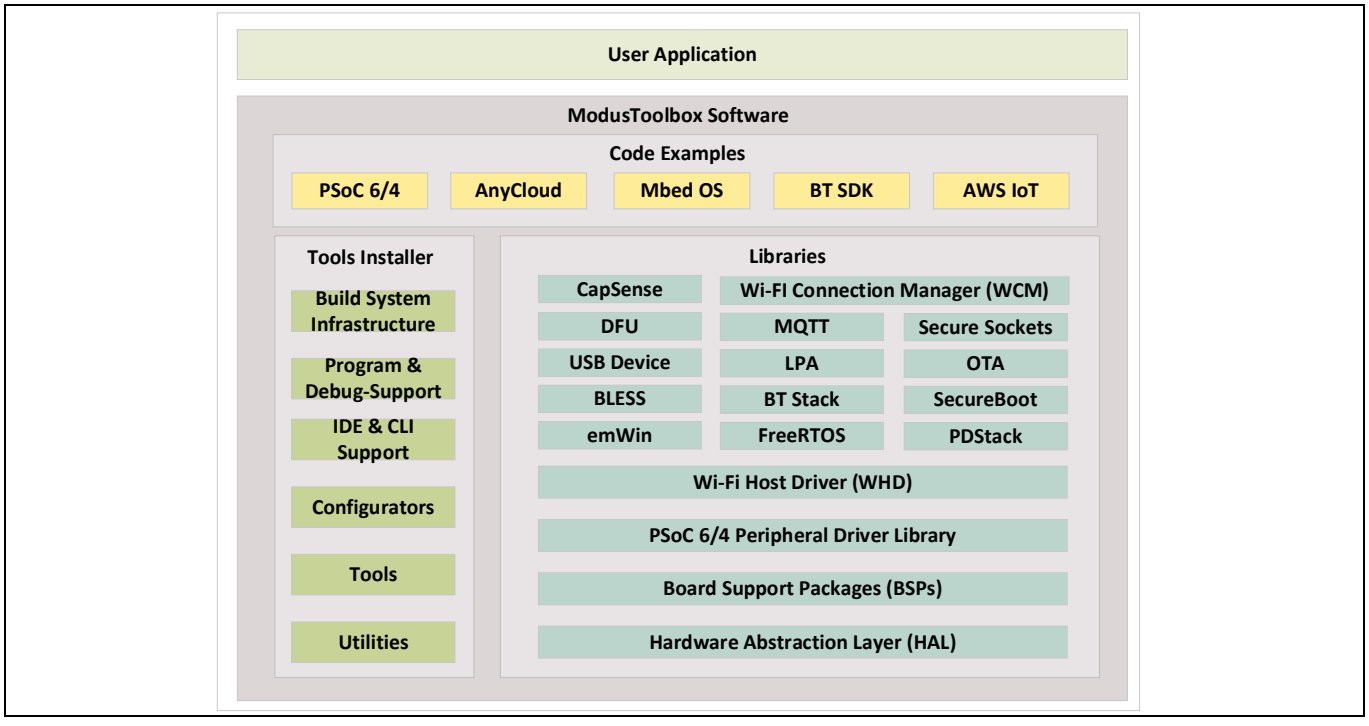


Figure 5 ModusToolbox™软件

开发

嵌入式 PMG1 MCU 是基于 Eclipse IDE 中的 ModusToolbox™软件所支持的原生开发流程。您可以在裸机或您所选的 RTOS 上使用它。

下面内容介绍的是 PMG1 MCU 的低层级资源。

1. **板级支持包 (BSP)：** BSP 是包含了特定于电路板的驱动程序和其它功能的固件层。板级支持包是一组库，它提供的 API 用于初始化电路板并对板级外设进行访问。该支持包包含用于 PMG1 MCU 的低级资源，例如外设驱动程序库 (PDL)，并具有用于电路板外设的宏。可以创建自定义 BSP 以支持终端应用电路板。有关更多信息，请参阅 **ModusToolbox®用户指南**中**板级支持包**一节的内容。
2. **外设驱动程序库 (PDL)：** PDL 在单个包中集成了器件头文件、启动代码以及外设驱动程序。驱动程序将硬件功能转为一组易于使用的 API。这样就降低了对寄存器使用和位结构深入了解的需要，从而简化了 PMG1 MCU 系列中大量外设的软件开发。您为应用程序配置驱动程序，然后通过调用 API 来初始化和使用外设。对于 PMG1 MCU 解决方案，片上模块 (包括 USBPD 和 SCB 在内) 由 MTB CAT2 PDL 进行控制。
3. **中间件：** 中间件是一组固件模块，可为应用程序提供特定功能。PDStack 是可用于 PMG1 MCU 的中间件，它包括 Type-C 和 USB-PD 状态机实现、PD 协议和 PD 策略管理器模块。

GitHub 存储库中以库的形式提供了所有低级资源。

Figure 6 显示了与 PMG1 MCU 相关的 ModusToolbox™软件模块。

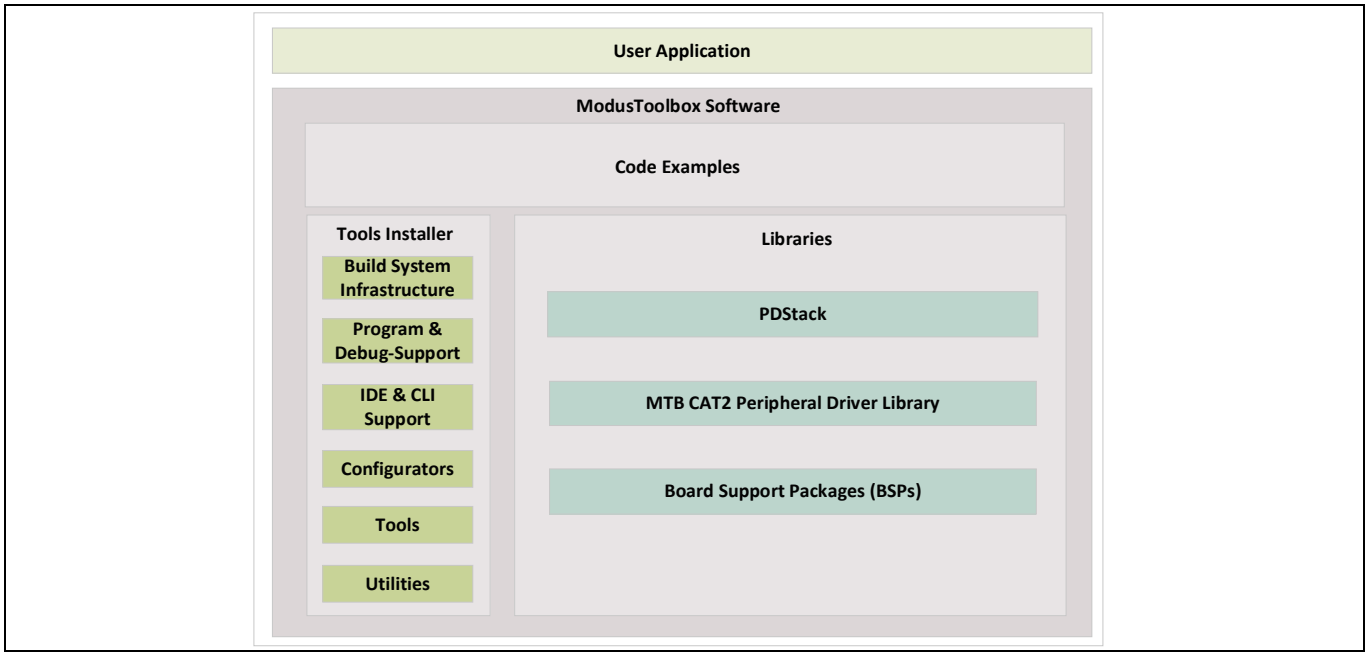


Figure 6　用于 PMG1 MCU 的 ModusToolbox™软件

2.2.1　PMG1 MCU 的软件资源

PMG1 MCU 的软件资源包括配置器、驱动程序、库和中间件，以及 Makefile 和脚本，有助于您入手开发 PMG1 MCU 的固件。

2.2.1.1 配置

ModusToolbox™软件提供了名为配置器的图形应用程序。使用该应用程序可以更加轻松地配置硬件模块或中间件。例如，您不再需要搜索所有文档才能将某个串行通信模块配置为具有所需配置的 UART，而只要打开配置器并设置波特率、奇偶校验和停止位。保存硬件配置后，该工具将生成 C 代码，从而使用所需配置来初始化硬件。

配置器彼此独立，但可以一起使用它们，从而获得灵活的配置选项。配置器可以独立使用，也可以与其它工具结合使用，或者在完整的 IDE 中使用。配置器用于：

- 设置选项并生成代码以配置驱动程序
- 为某个外设设置引脚和时钟等连接
- 设置选项并生成代码以配置中间件

对于 PMG1 MCU 应用，可用的配置器包括：

- **器件配置器：**设置系统 (平台) 功能以及基本外设 (例如，UART、定时器、PWM) 。
- **EZ-PD™配置器：**提供一个用户友好的工具，用于选择 PDStack 中间件的性能和配置参数以生成所需代码。

这些配置器创建自己的文件 (例如，用于 EZ-PD™配置器的 *design.mtbezpd*) 。通常在 BSP 中提供配置器文件 (*design.modus)* ，并且在代码示例中提供 *design.mtbezpd*。当根据 BSP 创建了某个应用程序时，这些文件将被复制到该应用程序中。您还可以为应用程序创建自定义器件配置器文件并将其覆盖 BSP 所提供的文件。更多有关信息，请参阅 **ModusToolbox™工具**网页中文档一节的内容。您也可以通过下面路径在 ModusToolbox™中查看这些文档：**帮助菜单 > ModusToolbox™通用文档 > ModusToolbox™ 文档索引**。

2.2.1.2 PMG1 MCU 的软件开发

英飞凌为客户提供了源代码和工具，用于进行 PMG1 MCU 的软件开发。您可以使用工具来指定您想要配置硬件的方式、生成代码 (为实现您在固件中所使用的目的) ，并提供用于附加功能的各种中间件库，例如 PDStack。通过使用该源代码，可以更轻松地为支持器件开发固件。这样您无需了解寄存器集仍能够快速地自定义和构建固件。

如 **Figure 7** 所示，通过 ModusToolbox™的 Eclipse IDE，您可以进行以下操作：

1. 选择板级支持包 (BSP) 。
2. 根据模板应用程序列表创建一个新应用程序，按套件过滤。
3. 添加中间件。
4. 使用 PDL 开发您的应用程序固件。

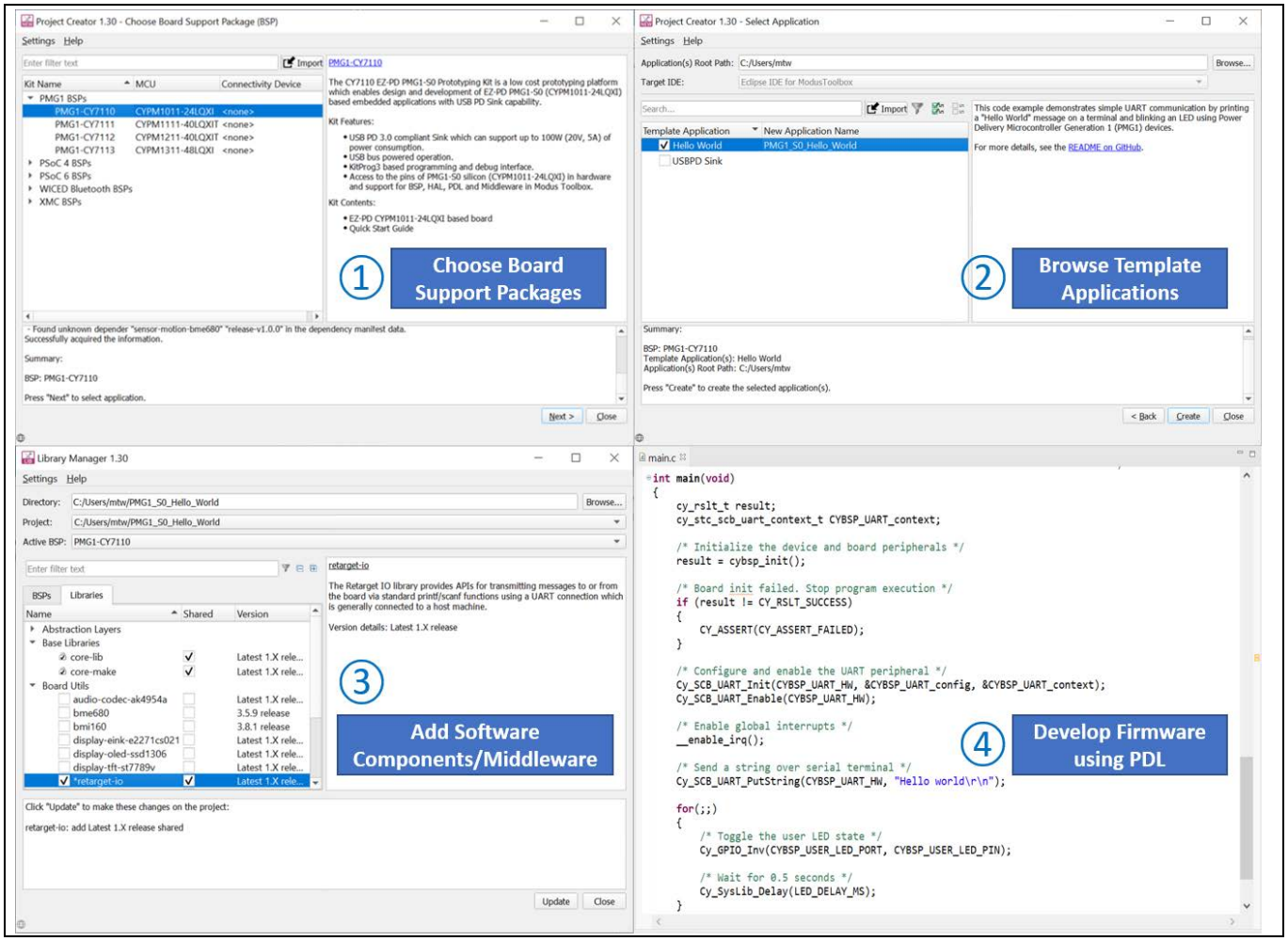


Figure 7　用于 ModusToolbox™软件资源和中间件的 Eclipse IDE

2.3　ModusToolbox™软件帮助

访问 **ModusToolbox**™软件主页以下载并安装最新版本的 ModusToolbox™软件。启动 ModusToolbox™软件的 Eclipse IDE，并指向以下路径以取得 ModusToolbox™的帮助。

依次选择**帮助 > ModusToolbox™通用文档或帮助 > ModusToolbox™的 Eclipse IDE 文档**

2.4　其它 IDE 的支持

除了 Eclipse IDE 以外，您还可以使用您喜欢的 IDE (例如 IAR Embedded Workbench 或 Visual Studio Code) 为 PMG1 MCU 开发固件。

有关更多详细信息，请参阅 ModusToolbox™软件用户指南中的“导出到 IDE”部分。英飞凌建议您使用 ModusToolbox™软件中提供的配置工具来生成资源配置。

2.5　编程与调试

ModusToolbox™软件的 Eclipse IDE 支持 KitProg3 并使用开放式片上调试器 (**OpenOCD**) 协议来调试 PMG1 MCU 应用程序。它还支持使用行业标准探针 (如 **Segger J-Link**) 的 GDB (GNU 调试器) 调试。

所有 PMG1 MCU 套件都有一个 KitProg3 板载编程器/调试器。它支持 Cortex®微控制器软件接口标准 – 调试访问端口 (CMSIS-DAP)。有关详细信息，请参阅 **KitProg3 用户指南**。

开发

更多有关使用 ModusToolbox™软件在 PMG1 器件上编程/调试固件的信息，请参阅 **ModusToolbox™用户指南**中的“编程和调试支持”部分。

2.6 PMG1 MCU 套件

Table 2 介绍了 PMG1 MCU 套件和相应 BSP 的列表。

Table 2 用于 PMG1 MCU 套件的 BSP

PMG1 MCU	PMG1 MCU 套件	BSP
CYPM1011-24LQXI	CY7110 EZ-PD™ PMG1-S0 原型开发套件	PMG1-CY7110
CYPM1111-40LQXI	CY7111 EZ-PD™ PMG1-S1 原型开发套件	PMG1-CY7111
CYPM1211-40LQXI	CY7112 EZ-PD™ PMG1-S2 原型开发套件	PMG1-CY7112
CYPM1311-48LQXI	CY7113 EZ-PD™ PMG1-S3 原型开发套件	PMG1-CY7113
CYPM1322-97BZXI	不适用	CYPM1322-97BZXI

3 我的第一个基于 Eclipse IDE 的 Modus Toolbox™软件设计

该章节包括以下内容：

- 说明了如何创建一个简单的、基于 PMG1 MCU 的设计，并将其构建和编程到原型开发套件上。
- 轻松学习 PMG1 MCU 设计技术以及使用 ModusToolbox™软件的 Eclipse IDE 的方法。

3.1 先决条件

在开始之前，请确保您已经拥有适合您的 PMG1 MCU 产品线的套件，并已安装好来自 **PMG1 MCU 网页**中最新的 ModusToolbox™软件。您还需要通过访问互联网才能在项目创建期间获取 GitHub 存储库。

有关 PMG1 MCU 套件的列表，请参见 **Table 2**。

3.1.1 硬件

该设计是为 CY7110 EZ-PD™ PMG1-S0 原型开发套件 (PMG1-CY7110) 开发的。然而，您可以为其它原型开发套件构建项目。请参见**应用程序开发流程**部分内容。

3.1.2 软件

- ModusToolbox™软件 2.3 或更高版本

安装软件后，请参阅 **ModusToolbox™软件用户指南**以了解该软件。

3.2 应用程序开发流程

这些说明分为几个部分。每个部分专门介绍了应用程序开发工作流程的某个阶段。主要部分包括：

- **第一部分：创建新的应用**
- **第二部分：查看和修改设计**
- **第三部分：编写固件**
- **第四部分：构建应用程序**
- **第五部分：编程器件**
- **第六部分：测试您的设计**
- **第七部分：调试应用程序**

已在 **Table 2** 中列出的所有 PMG1 MCU 套件上测试下面内容所介绍的代码。

3.3 设计概述

该设计使用 PMG1-S0 MCU 执行两个任务：UART 通信和 LED 控制。该项目使用 UART 将“Hello World” (你好世界) 消息打印到串行端口流，并使套件上用户的 LED 闪烁发亮。

3.4 实现流程

3.4.1 第一部分：创建新的应用

本节逐步向您介绍了创建新应用程序的流程。该流程使用“Hello World” (你好世界) 代码示例并指导您完成设计开发阶段和编程。

如果您熟悉使用 ModusToolbox™软件开发项目，则可以直接使用代码示例。它是一个完整的设计，包含为支持的套件所编写的全部固件。您可以浏览代码中的指令并观察这些步骤实现的流程。

如果您从头开始并按照本节中的所有说明进行操作，则在按照说明进行操作时可以使用代码示例作为参考。

1. 启动 ModusToolbox™软件的 Eclipse IDE。

   启动 **ModusToolbox™软件版本 2.3 的 Eclipse IDE**，从而开始使用。请注意，需要保持网络连接，ModusToolbox™软件的 Eclipse IDE 才能将模板应用程序成功克隆到您的机器上。

2. 选择一个配置文件。

   在启动时，ModusToolbox™的 Eclipse IDE 将显示一个对话框，用于选择作为工作区目录的目录。工作区目录用于存储工作区首选项和开发工件。您可以通过点击 **Browse** 按钮来选择一个现有的空白目录，如 **Figure 8** 所示。或者，您可以输入一个作为工作区目录使用的目录名称以及完整的路径，Eclipse IDE 将为您创建该目录。

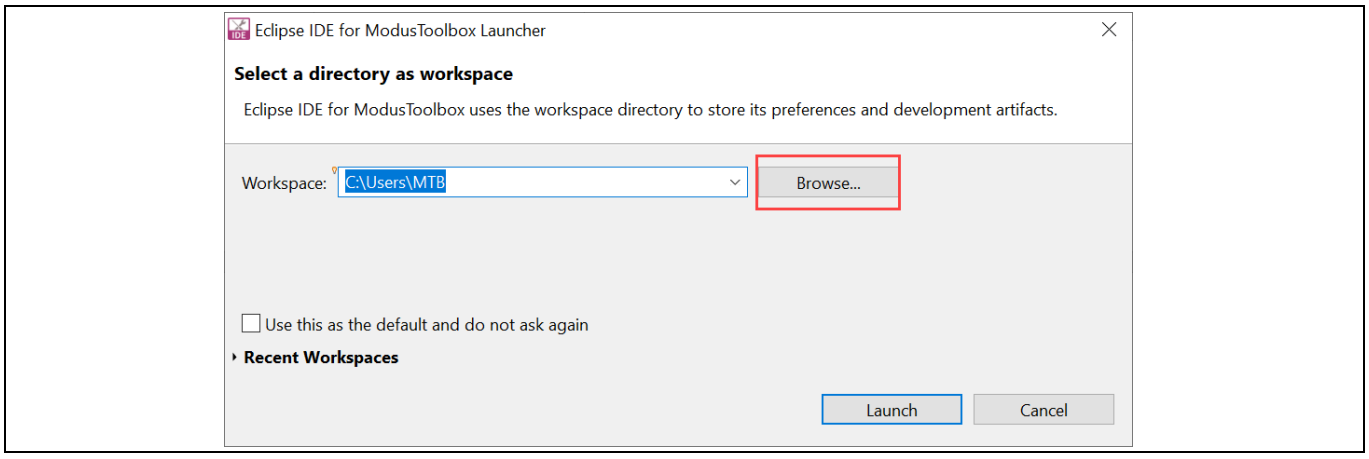


Figure 8 为工作区选择/创建一个目录

3. 创建一个新的 ModusToolbox™应用
   a) 在 Quick Panel 中 Start 组下点击 New Application 项。
   b) 或者，依次选择 **File > New > ModusToolbox™ Application**，如 **Figure 9** 所示。

      Eclipse IDE for ModusToolbox™ Application 窗口将会出现。

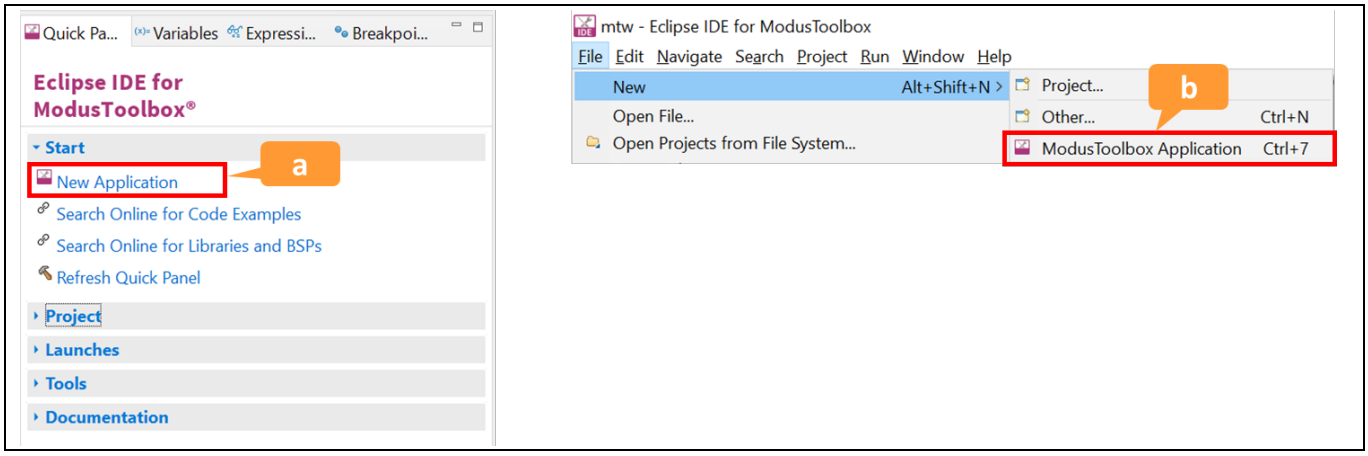


Figure 9　创建一个新的 ModusToolbox™应用

4. 选择一个目标 PMG1 MCU 套件。

   a) 在 **Choose Board Support Package (BSP)** 对话框中，选择您所拥有的**套件名称**。由于要创建 PMG1-S0 的应用，所以请选择 PMG1-CY7110。有关该步骤的帮助内容，请参见 **Figure 10**。

   b) 点击 **Next** 按钮。

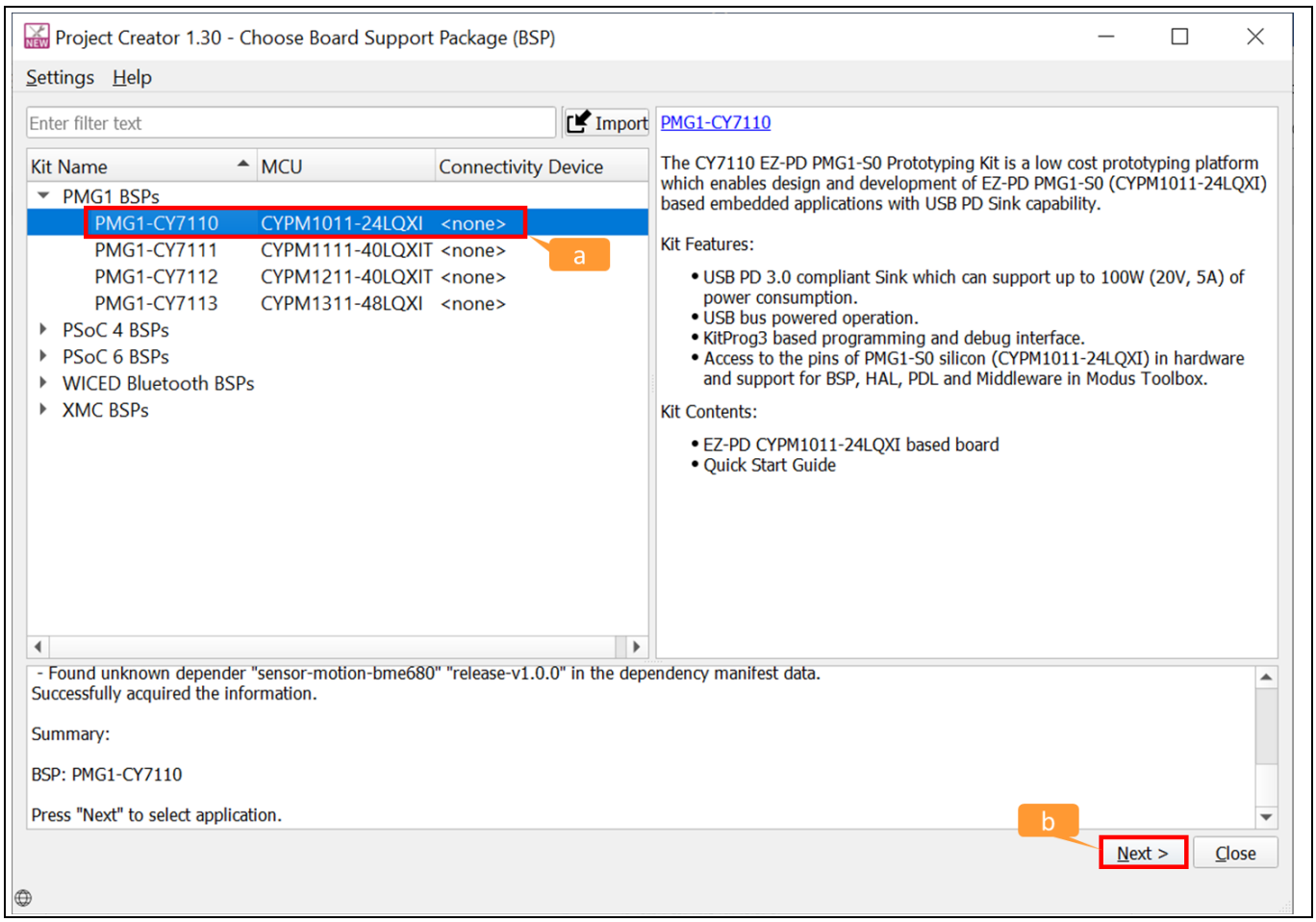


Figure 10　选择目标硬件

c) 在 **Select Application** 对话框中，选择 **Hello World** 模板应用程序，如 **Figure 11** 所示。

d) 在 **Name** 字段中，输入应用的名称，例如 **PMG1_S0_Hello_World**。您也可以选择使用默认名称。

e) 点击 **Create** 以创建应用程序，并等待项目创建器在项目创建成功后自动关闭。

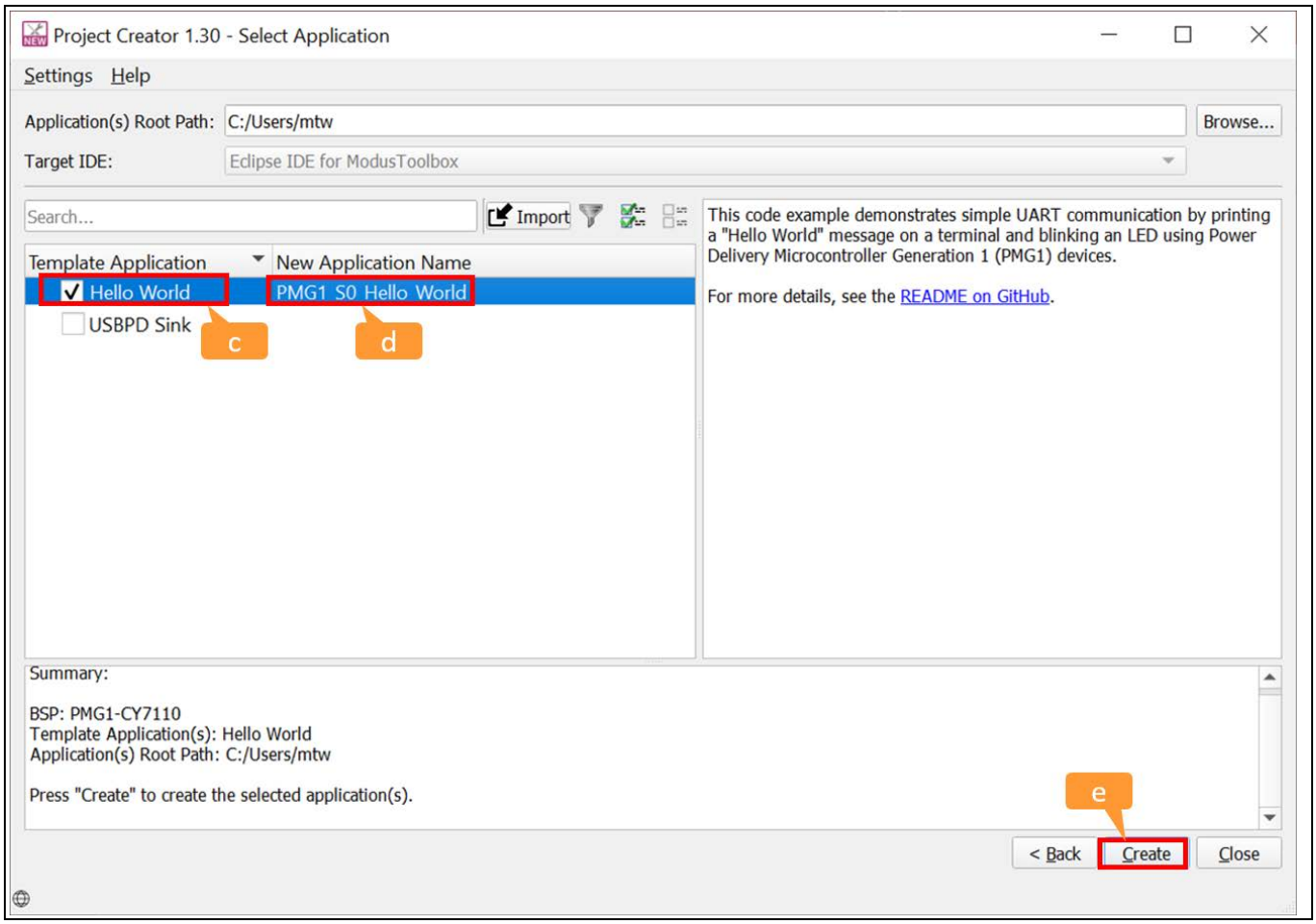


Figure 11　选择模板应用程序

您已成功为 PMG1-S0 MCU 创建了一个新的 ModusToolbox™软件应用程序。BSP 在 CY7110 EZ-PD™ PMG1-S0 原型开发套件 (PMG1-CY7110) 上使用了 CYPM1011-24LQXI 器件。

成功创建好某个应用后会打开 *README.md* 文件，该文件包含模板应用实现的信息。

如果您使用基于 PMG1-S0 MCU 或其它 PMG1 MCU 器件编号的自定义硬件，请参阅 **ModusToolbox™软件用户指南** (**帮助** > **ModusToolbox™通用文档**) 中“创建您的 BSP”一节。

3.4.2　第二部分：查看和修改设计

Figure 12 介绍了 ModusToolbox™软件项目浏览器界面，该界面用于显示应用程序项目的结构。

在 ModusToolbox™软件的 Eclipse IDE 中，PMG1 MCU 应用程序包含的项目可用于为 CM0/CM0+ CPU 开发代码。项目文件夹由多个子文件夹组成 – 每个子文件夹表示项目的一个特定方面。

a) 应用程序项目包含一个 Makefile，它通常位于根文件夹中。它提供了关于如何重新创建项目的说明。该文件还包含 make 工具所要使用的指令集，从而使用这些指令来编译和链接应用程序项目。一个应用程序中可以有多个项目； 每个依赖项目通常位于应用程序文件夹中自己的文件夹内，并具有自己的 Makefile。

b) *build* 文件夹包含由项目的 make 构建产生的所有工件。输出文件由目标 BSP 组织。

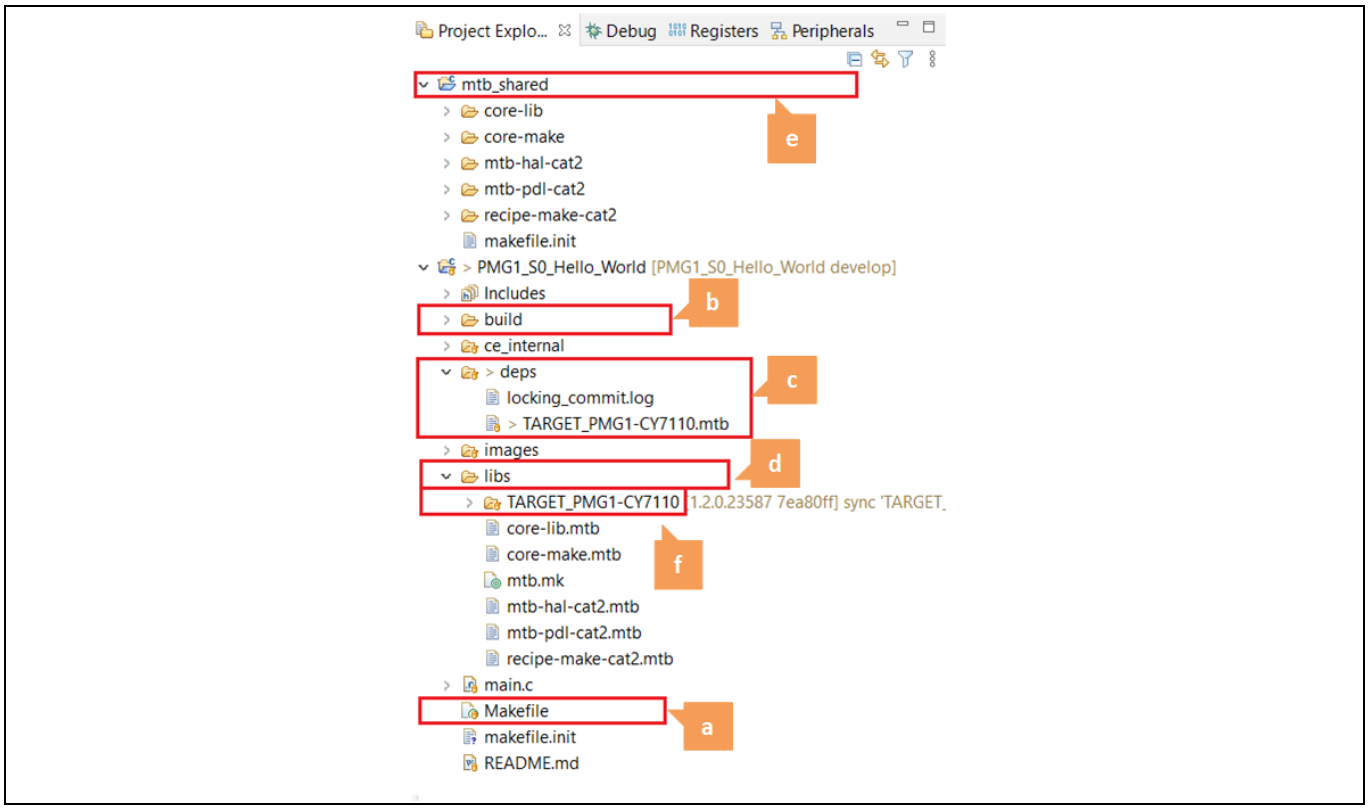


Figure 12　项目资源管理器

c) *deps* 文件夹包含*.mtb* 文件，这些文件提供了 ModusToolbox™软件要提取的 BSP/库所在的位置，该 BSP/库被应用程序直接引用。这些文件通常包含整个库的 GitHub 位置。*.mtb* 文件还包含一个 git Commit Hash (提交哈希) 或 Tag (标记)，用于说明要获取库的版本以及用于存储该库的位置的路径。

d) *libs* 文件夹还包含*.mtb* 文件，这些文件指向间接包含的库，它们作为 BSP 或其它库的依赖项。对于每个间接依赖项，库管理器将在该文件夹中放置一个*.mtb* 文件。根据 *deps* 文件夹中可用的目标来填充这些文件。

例如，使用 BSP lib 文件 *TARGET_PMG1-CY7110.mtb* 时，以下*.mtb* 文件便被添加到 *libs* 文件夹中：**core-lib.mtb**、**core-make.mtb**、**mtb-hal-cat2.mtb**、**mtb-pdl-cat2.mtb**、**recipe-make-cat2.mtb**。

libs 文件夹包含 *mtb.mk* 文件，该文件存储应用程序所需的所有库的相对路径。构建系统使用该文件查找应用程序所需的所有库。

e) 默认情况下，在创建新应用程序或向现有应用程序中添加库并将其指定为共享时，所有库将被放置在与应用程序目录相邻的 *mtb_shared* 目录中。

使用相同版本的 BSP/库的不同应用程序共享 *mtb_shared* 文件夹。

f) *TARGET_x* 文件夹下列出了 BSP 所提供的文件。BSP 的 *GeneratedSource* 文件夹包含了由器件和外设配置器所生成的全部配置文件，这些文件都以 *cycfg_*为前缀。这些文件包含 BSP 定义的设计配置。通过点击 Quick Panel 中的 **Device Configurator** 链接，您可以查看和修改设计配置。但要注意，如果您将 BSP 库升级到更新版本，则对 *design.x* 文件所做的手动编辑将被丢失。

关注的内容是 *COMPONENT_BSP_DESIGN_MODUS* 文件夹中的配置文件。点击 Quick Panel 中的 **Device Configurator** 链接。**Figure 13** 显示了名称为 **Device Configurator** 的结果窗口。您也可以通过双击打开其它 *design.x* 文件，或者点击 **Quick Panel** 中的相应链接来打开相应的配置器。

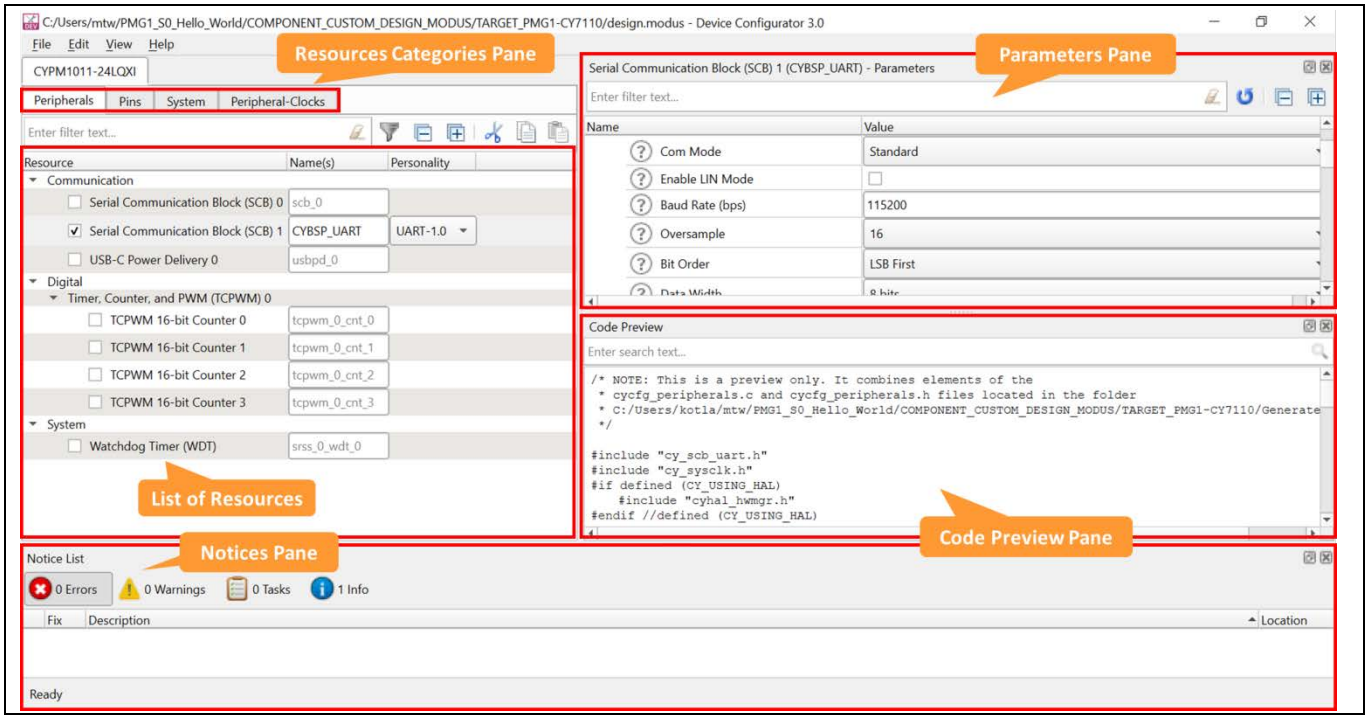


Figure 13 design.modus 概述

Device Configurator 窗口提供了 **Resources Categories** 面板。在该面板中，您可以从**资源列表**中选择可用于器件的不同资源，例如外设、引脚和时钟。

通过选择资源的**性能**，您可以指定资源的行为。例如，**串行通信模块 (SCB)** 资源可以具有 **EZI2C**、**I2C**、**SPI** 或 **UART** 的性能。其**名称**是您在开发固件过程中所使用的资源名称。可以指定一个或多个名称，并使用逗号进行分隔 (没有空格)。

在 **Parameters** 面板中，您可以为每个启用的资源和选定的性能输入配置参数。**Code Preview** 面板显示了根据所选配置参数而生成的配置代码。在 *GeneratedSource* 文件夹中的 *cycfg_*文件内填充该代码。由配置引起的任何错误、警告和信息等内容都会显示在 **Notices** 面板中。

在开发过程的这个阶段中，您便可以将任何所需的中间件添加到设计中。**Figure 27** 显示的是一个添加中间件的示例。对于“Hello World”示例项目，不需要添加任何额外中间件。

通过使用 BSP 提供的函数 API，可以在代码中直接对调试 UART 外设、引脚和系统时钟进行配置。请参见**第三部分：编写固件**。

3.4.3 第三部分：编写固件

在开发过程的这一步，您已经通过模板应用程序创建了一个应用程序。在本节中，您将编写用于实现设计功能的固件。

如果您是从头开始操作，您可以将本节所提供的源代码复制到应用程序项目的 *main.c* 文件中。如果您使用的是 PMG1 MCU “Hello World”代码示例，则所有需要的文件都已经存在于应用程序中。

3.4.3.1 固件流程

现在您可以检查应用程序中 *main.c* 文件内的代码。**Figure 14** 显示了固件流程图。

该示例的资源初始化由 CM0 CPU 执行。它将配置系统时钟、引脚、时钟到外设的连接以及其它平台资源。

我的第一个基于 Eclipse IDE 的 Modus Toolbox™软件设计

当 CM0 CPU 被启用时，时钟和系统资源将通过 BSP 初始化函数得到初始化。UART 外设被配置并启用。它在终端仿真器上打印出“Hello World”消息。CPU 以 500 毫秒的延迟持续切换 LED 状态。

请注意，应用程序代码使用 BSP/中间件函数来执行预期功能。

将以下代码片段复制到应用程序项目的 *main.c* 文件中。

```
/*******************************************************************************
 * Include header files
 ******************************************************************************/
#include "cy_pdl.h"
#include "cybsp.h"


/*******************************************************************************
* Macros
*******************************************************************************/
#define LED_DELAY_MS               (500u)
#define CY_ASSERT_FAILED           (0u)


* Function Name: main
int main(void)
{
    cy_rslt_t result;
    cy_stc_scb_uart_context_t CYBSP_UART_context;

    /* Initialize the device and board peripherals */
    result = cybsp_init();

    /* Board init failed. Stop program execution */
    if (result != CY_RSLT_SUCCESS)
    {
        CY_ASSERT(CY_ASSERT_FAILED);
    }

    /* Configure and enable the UART peripheral */
    Cy_SCB_UART_Init(CYBSP_UART_HW, &CYBSP_UART_config, &CYBSP_UART_context);
    Cy_SCB_UART_Enable(CYBSP_UART_HW);

    /* Enable global interrupts */
    __enable_irq();

    /* Send a string over serial terminal */
    Cy_SCB_UART_PutString(CYBSP_UART_HW, "Hello world\r\n");

    for(;;)
    {
        /* Toggle the user LED state */
        Cy_GPIO_Inv(CYBSP_FW_LED_PORT, CYBSP_FW_LED_PIN);

        /* Wait for 0.5 seconds */
        Cy_SysLib_Delay(LED_DELAY_MS);
    }
}

/* [] END OF FILE */
```

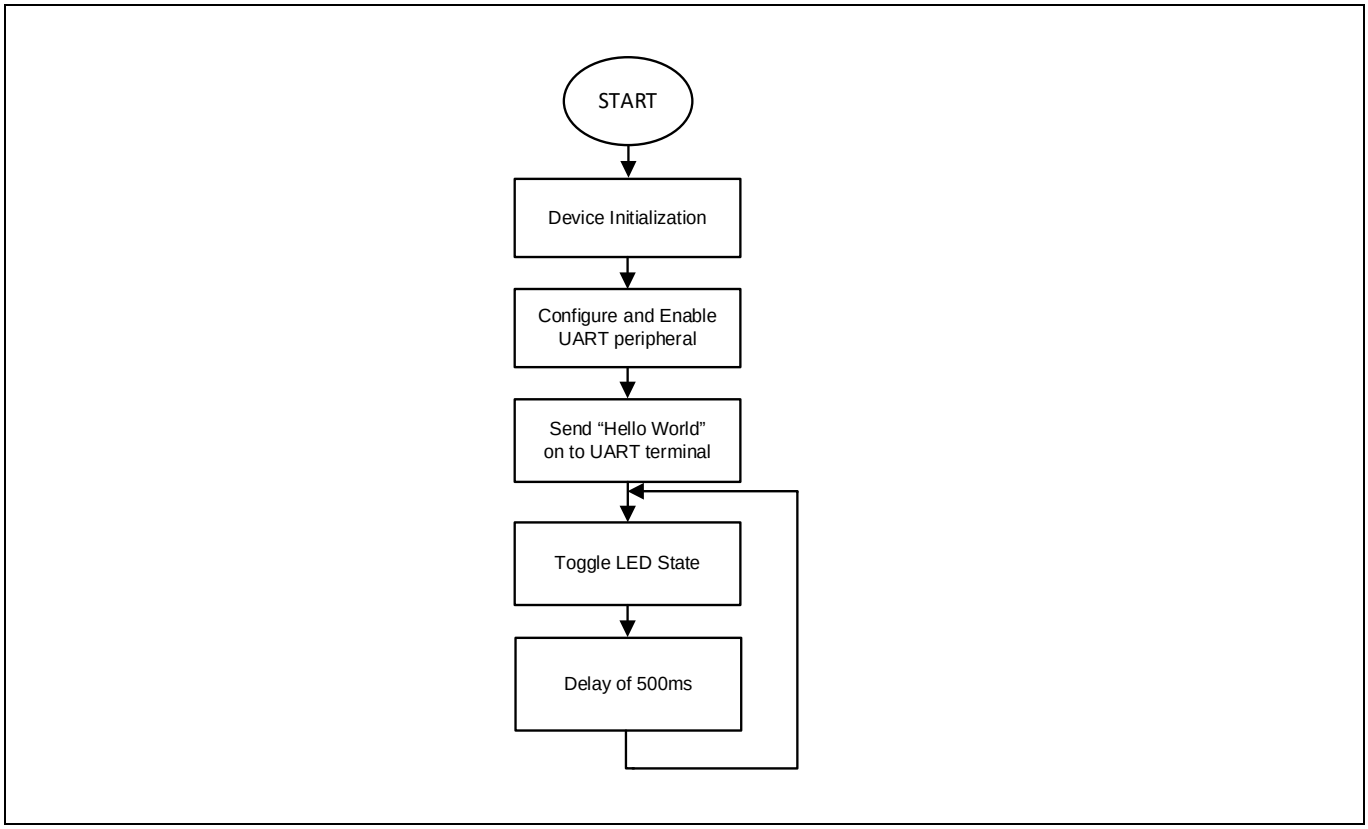


Figure 14　流程图

该图总结了固件在代码示例中的工作原理。请随意浏览源文件以获得更深入的理解。

3.4.4　第四部分：构建应用程序

本节阐述了构建应用程序的方法。

1. 构建应用程序。
   a) 在 Project Explorer 窗口中选择应用程序项目，并点击 Quick Panel 中**<名称>**组下方的 **Build <名称> Application** 快捷指令。该操作将选择 **Debug** 构建配置，并编译/链接构成应用程序的所有项目。
   b) **Console** 视图列出了构建操作的结果，如 **Figure 15** 所示。

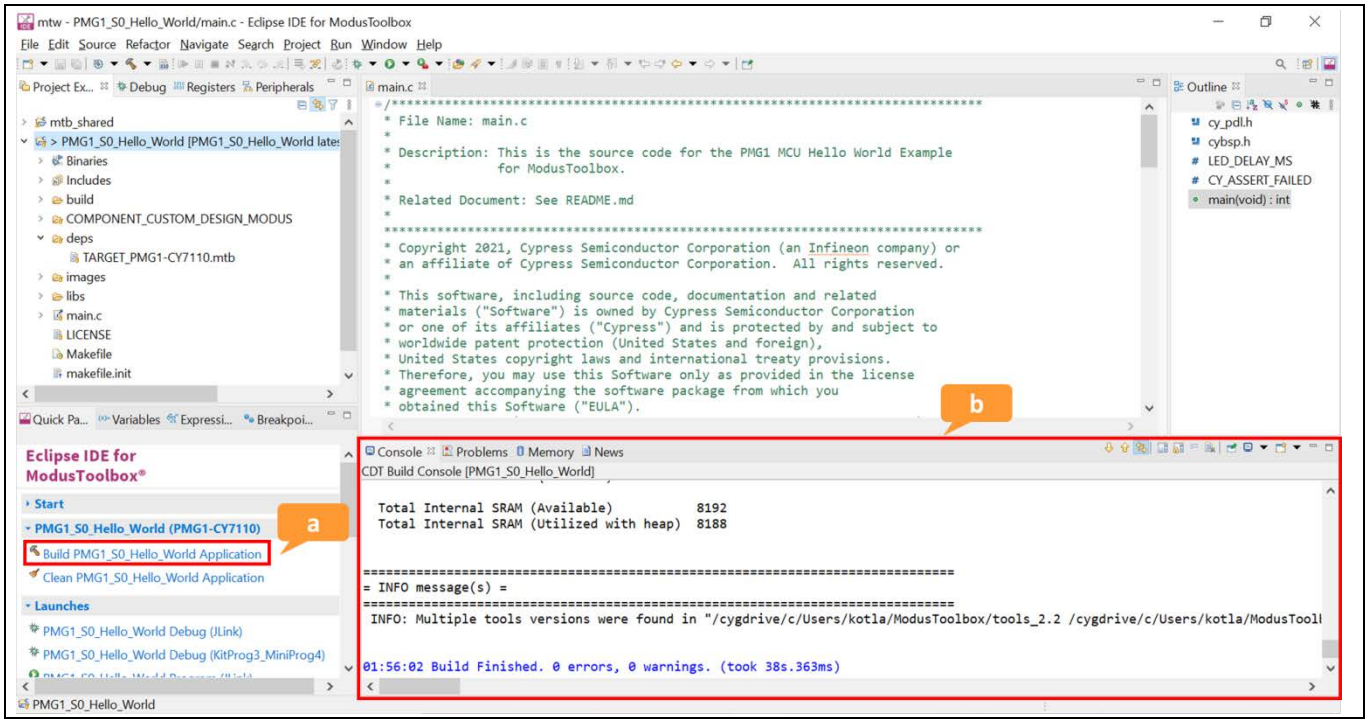


Figure 15 应用程序

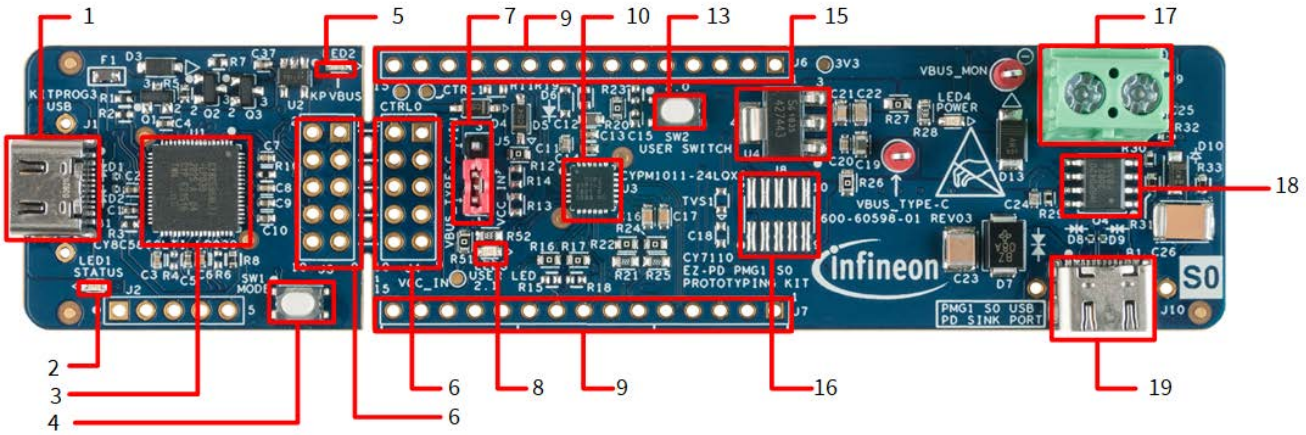
如果遇到错误，请尝试以下选项：

- 重新查看之前的步骤以确保您完成了所有需要的任务。
- 检查 **Problems** 选项卡以查找有关错误的更多详细信息，并解决这些错误。
- 在 Quick Panel 中，点击‘Clean <应用程序名称>’项，然后重新尝试构建项目。

Note: *您还可以使用指令行接口(CLI)来构建应用程序。请参阅* ***ModusToolbox™用户指南*** *中* ***使用指令行*** *一节。您也可以通过下面路径在 ModusToolbox™软件中查看该文档：* ***帮助菜单 > ModusToolbox™通用文档*** *。*

3.4.5 第五部分：编程器件

本节介绍了编程器件的方式。有关 CY7110 EZ-PD™ PMG1-S0 原型开发套件的信息，请参见 **Figure 16**。请参见 **PMG1 MCU 网页**中的“套件”一节。



1. KitProg3 USB Type-C 端口 (J1)
2. KitProg3 状态 LED (LED1)
3. PSoC™ 5LP 控制器 (U1)
4. KitProg3 模式切换 (SW1)
5. KitProg3 VBUS LED (LED2)
6. KitProg3 插头 (J3, J4) *
7. 电源选择跳线器 (J5)
8. 用户 LED (LED3)
9. I/O 插头 (J6, J7) *
10. CYPM1011-24LQXI (PMG1-S0)
13. 用户开关 (SW2)
15. 3.3V 板载 LDO
16. 10 引脚 SWD/JTAG 插头 *
17. DC_OUT 终端模块 (J9)
18. 加载开关
19. PMG1 MCU USB Type-C 端口 (J10)

*只有焊盘图形；没有贴装在电路板上。

Figure 16 选择 CY7110 EZ-PD™ PMG1-S0 原型开发套件

ModusToolbox™软件使用 OpenOCD 协议对 PMG1 MCU 器件上的应用程序进行编程和调试。为了让 ModusToolbox™软件能够识别套件上的器件，该套件必须运行 KitProg3。

如果您使用的是带有内置编程器的 PMG1 MCU 原型开发套件，请按照第一和第二步骤进行操作。

如果您在自己的硬件上进行开发，您可能需要一个硬件编程器/调试器； 例如，英飞凌 **CY8CKIT-005 MiniProg4**。

1. 在原型开发套件上进行以下硬件连接：
   a) 使用 Type-C 线缆将套件上的 J1 (KitProg3 USB Type-C 端口) 连接器连接到主机。
   b) 连接 J5 (电源选择跳线器) 引脚 2 和 3，以用于编程模式。

Note: 通过连接 J5 (电源选择跳线器) 引脚 1 和 2 (用于操作模式)，以及连接 Type-C 线缆 J1 和 J10，也可以对器件进行编程。

2. 对电路板进行编程：
   a) 在 ModusToolbox™软件中，选择 Project Explorer 中的应用程序项目。
   b) 在 Quick Panel 中，向下滚动，然后点击**\<应用程序名称> Program (KitProg3_MiniProg4)**。

IDE 将选择并运行适当的运行配置。请注意，如果自上次构建以后任何文件发生了变化，则将在该步骤内执行一次构建。

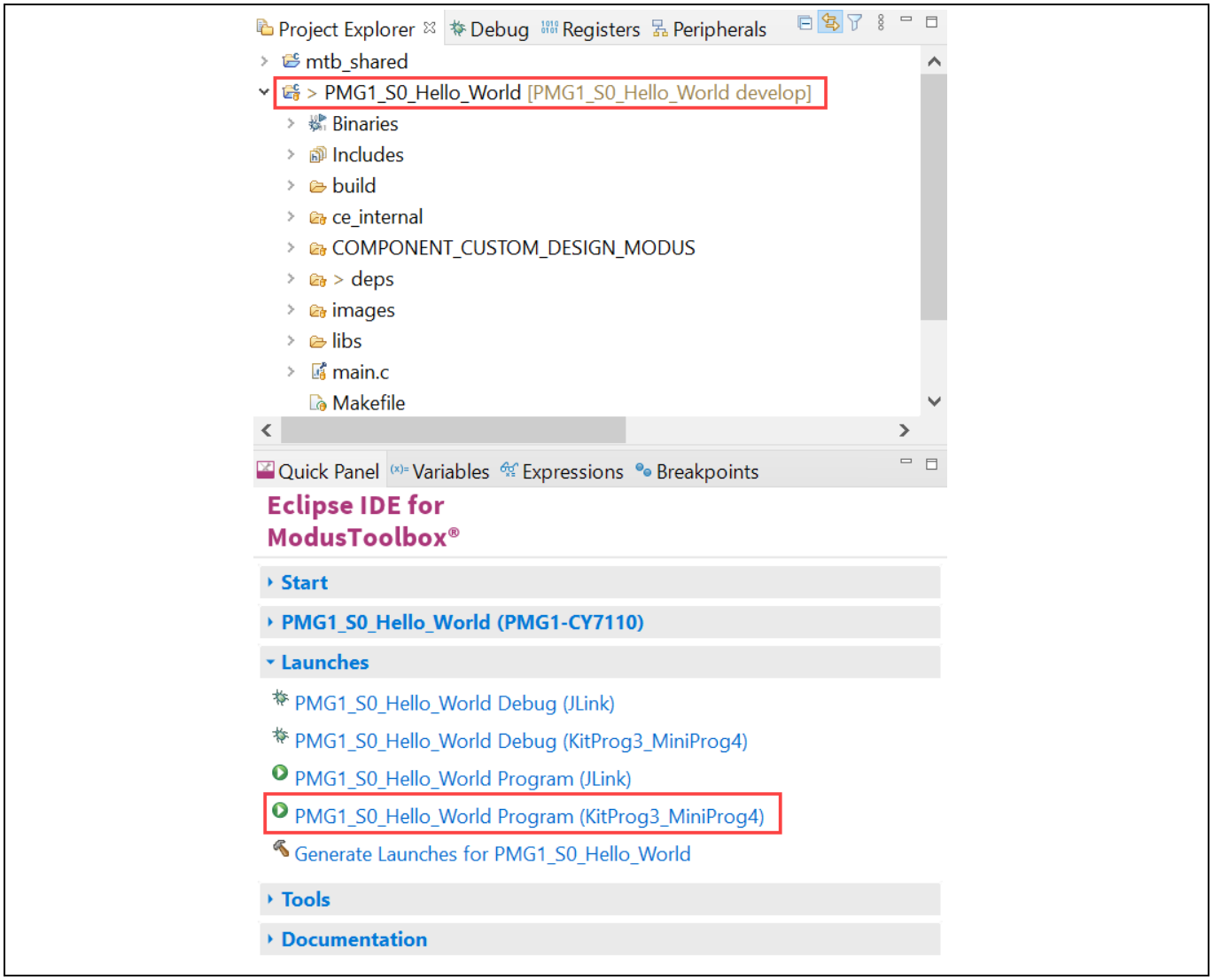


Figure 17　将应用程序编程到器件上

3. ModusToolbox™软件中的 **Console** 视图列出了编程操作的结果，如 **Figure 18** 所示。

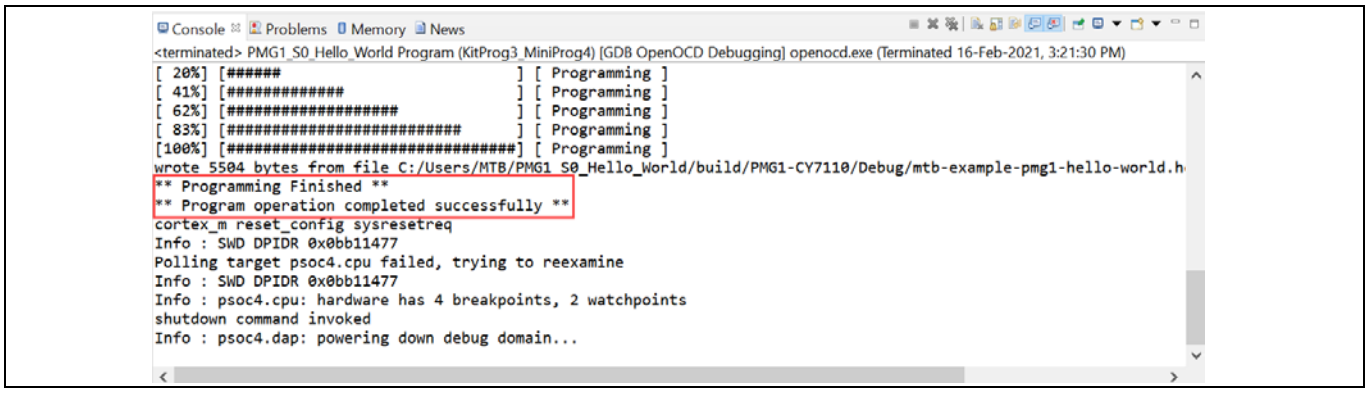


Figure 18　控制台–编程结果

3.4.6 第六部分：测试您的设计

本节介绍的是如何测试您的设计。

1. 需要进行以下各硬件连接：
   a) 连接 J5 (电源选择跳线器) 引脚 1 和 2，以用于操作模式。
   b) 将 J6.10 连接到 J3.8，并将 J6.9 连接到 J3.10，从而能够在 KitProg3 和 PMG1-S0 之间建立 UART 连接。
   c) 将 Type-C 线缆从套件上的 J1 (KitProg3 USB Type-C 端口) 连接器连接到主机。
   d) 确认 KitProg3 VBUS LED (LED2) 和状态 LED (LED1) 发出的是琥珀色光。
2. 打开终端程序并选择 KitProg3 COM 端口。在本应用笔记中，Tera Term 被作为 UART 终端仿真器使用，用于查看结果。您可以使用所选择的任何终端来查看结果。

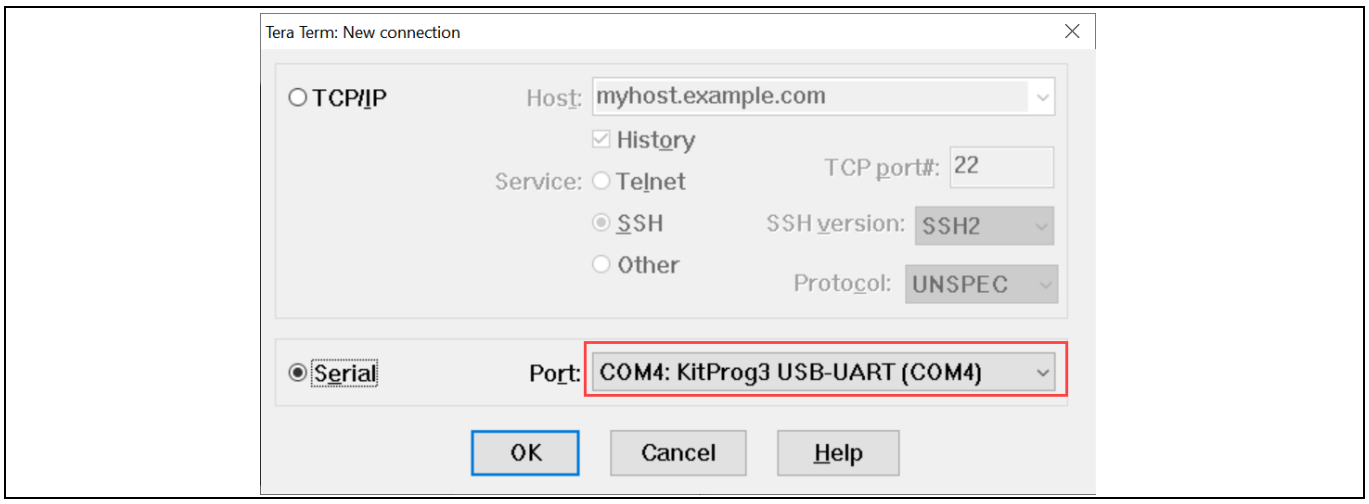


Figure 19 在 Tera Term 中选择 KitProg3 COM 端口

3. 将串行端口参数设置为 8N1 和 115200 波特率。

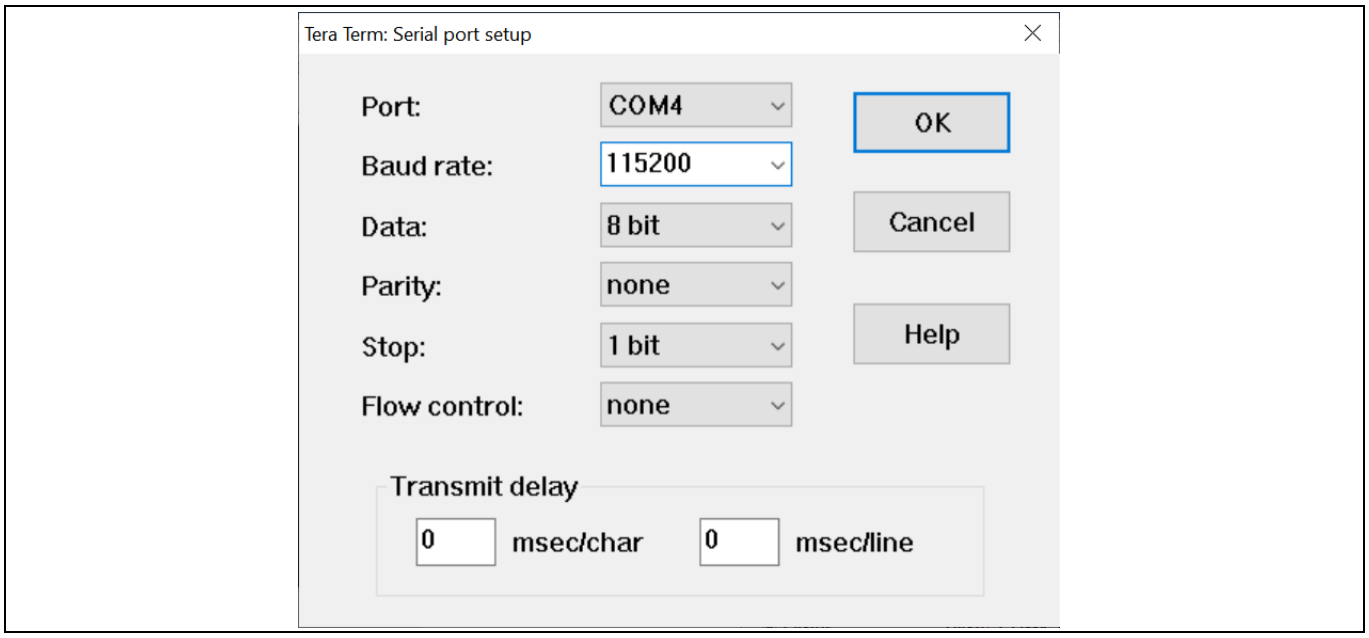


Figure 20 配置 Tera Term 中的波特率

4. 使用 Type-C 线缆将套件上的 J10 (PMG1 MCU USB Type-C 端口) 连接器连接到主机。
5. 确认 UART 终端上显示了“Hello World”。

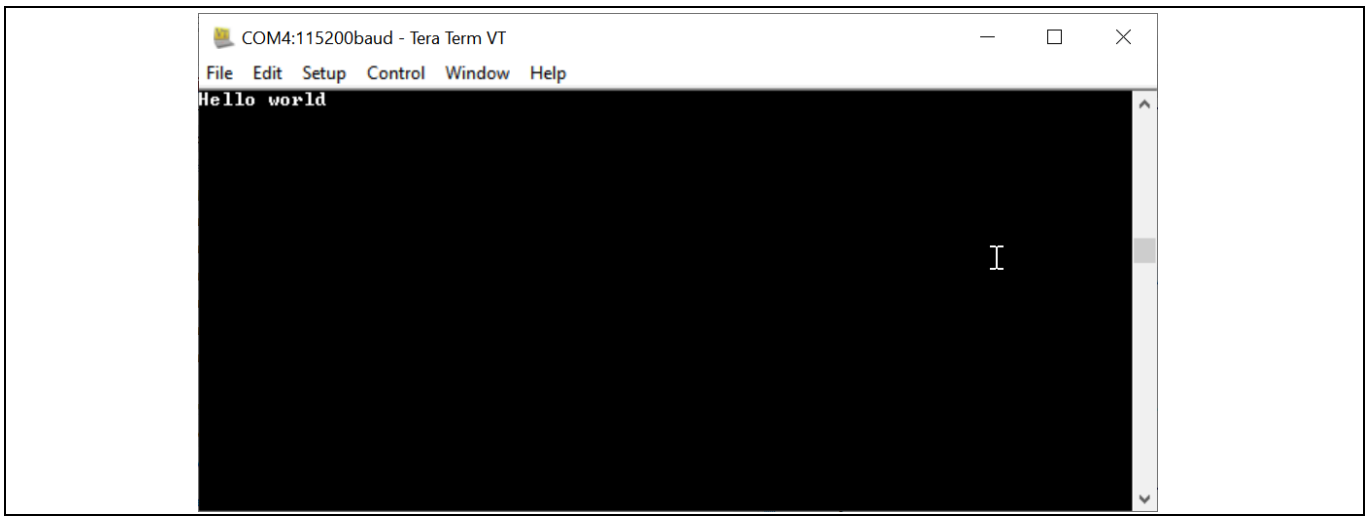


Figure 21 终端输出中的“Hello World”

6. 确认电源 LED (LED4) 发出绿光，用户 LED (LED3) 以绿色闪烁。

3.4.7 第七部分：调试应用程序

可以使用 ModusToolbox™软件的 Eclipse IDE 来调试应用程序。与编程模式相同，可以通过下面三种接口中的其中一种使原型开发套件进入调试模式：

- KitProg3 USB Type-C 端口
- 10 引脚 SWD 插头
- I/O 插头引脚

在开始调试之前，需要将该套件配置为操作模式。

3.4.7.1 通过 KitProg3 接口进行调试

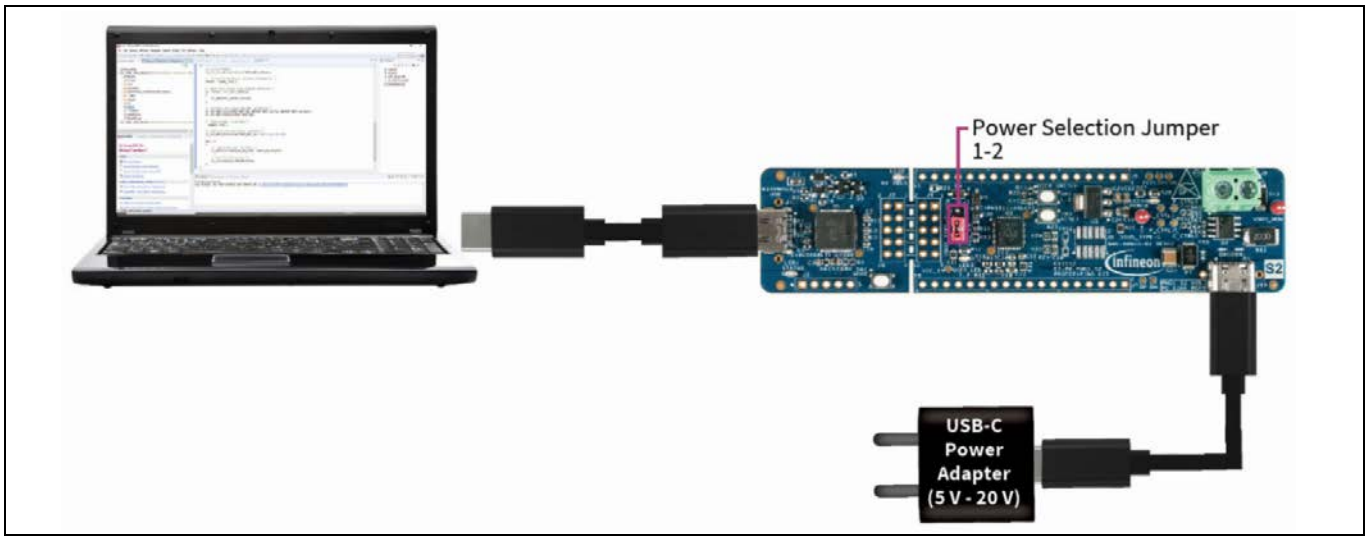


Figure 22 通过 KitProg3 进行调试

我的第一个基于 Eclipse IDE 的 Modus Toolbox™软件设计

为了在 ModusToolbox™软件的 Eclipse IDE 中通过 KitProg3 接口调试 CY7111 和 CY7112 原型开发套件上的应用程序，请执行以下各操作。

1. 打开 ModusToolbox™软件的 Eclipse IDE，然后在主机 PC 上选择 PMG1 MCU 项目。有关详细信息，请参阅 **ModusToolbox™软件的 Eclipse IDE 用户指南**。
2. 将跳线分流器连接在电源选择跳线器 (J5) 的引脚 1–2 上，以将套件配置为操作模式。
3. 将 USB-PD 接收源端口连接到 USB PD 发送源，以激活板载 LDO、负载开关和用户 LED。确保 LED4 (即电源 LED) 发出绿光。
4. 通过编程 KitProg3 USB Type-C 端口将该套件连接到主机 PC。确保 LED1 和 LED2 发出琥珀色光。LED2 (即 KitProg3 电源 LED) 表示 KitProg3 模块已通电。LED1 (即状态 LED) 指示了编程/调试模式和状态，并且在 KitProg3 通电时，它处于打开状态。
5. 跳转到 **Quick Panel** 选项卡。在 **Launches** 部分点击**<应用程序名称> Debug (KitProg3_MiniProg4)** 项。
6. IDE 将切换到调试模式，并在 main()函数的第一行处停止。这样便表示应用程序已准备好，可以进行调试。

为了在 ModusToolbox™软件的 Eclipse IDE 中通过 KitProg3 接口调试 CY7110 原型开发套件上的应用程序，请执行以下各操作。

1. 打开 ModusToolbox™软件的 Eclipse IDE，然后在主机 PC 或笔记本电脑上选择 PMG1 MCU 项目。有关详细信息，请参阅 **ModusToolbox™软件的 Eclipse IDE 用户指南**。
2. 需要将电源选择跳线 (J5) 分流器连接在电源选择跳线器 (J5) 的引脚 2–3 上，这样 ModusToolbox™软件可以使目标 (PMG1-S0) 进入电源循环模式。
3. 将 USB-PD 接收源端口连接到 USB PD 发送源，以激活板载 LDO、负载开关和用户 LED。确保 LED4 (即电源 LED) 发出绿光。
4. 通过编程 KitProg3 USB Type-C 端口将该套件连接到主机 PC。确保 LED1 和 LED2 发出琥珀色光。LED2 (即 KitProg3 电源 LED) 表示 KitProg3 模块已通电。LED1 (即状态 LED) 指示的是编程/调试模式和状态，并且在 KitProg3 通电时，它处于打开状态。
5. 跳转到 **Quick Panel** 选项卡。在 **Launches** 部分点击**<应用程序名称> Program (KitProg3_MiniProg4)** 项，从而将固件下载到套件中，如 **Figure 23** 所示。
6. 程序下载完成后，将跳线器更改为电源选择跳线器 (J5) 的引脚 1-2，以便将套件配置为操作模式，用于执行调试操作。
7. 跳转到 **Run** 菜单，然后点击 **Debug Configurations** 项，如 **Figure 24** 所示。
8. 在 GDB OpenOCD Debugging 项下，选择**<应用程序名称> Attach (KitProg3_MiniProg4)** 项，然后点击 **Debug** 按钮，如 **Figure 25** 所示。这样会启动附加到正在运行的目标的调试会话，而不需要进行编程或复位。更多有关调试配置的信息，请参阅 **ModusToolbox™用户指南**。

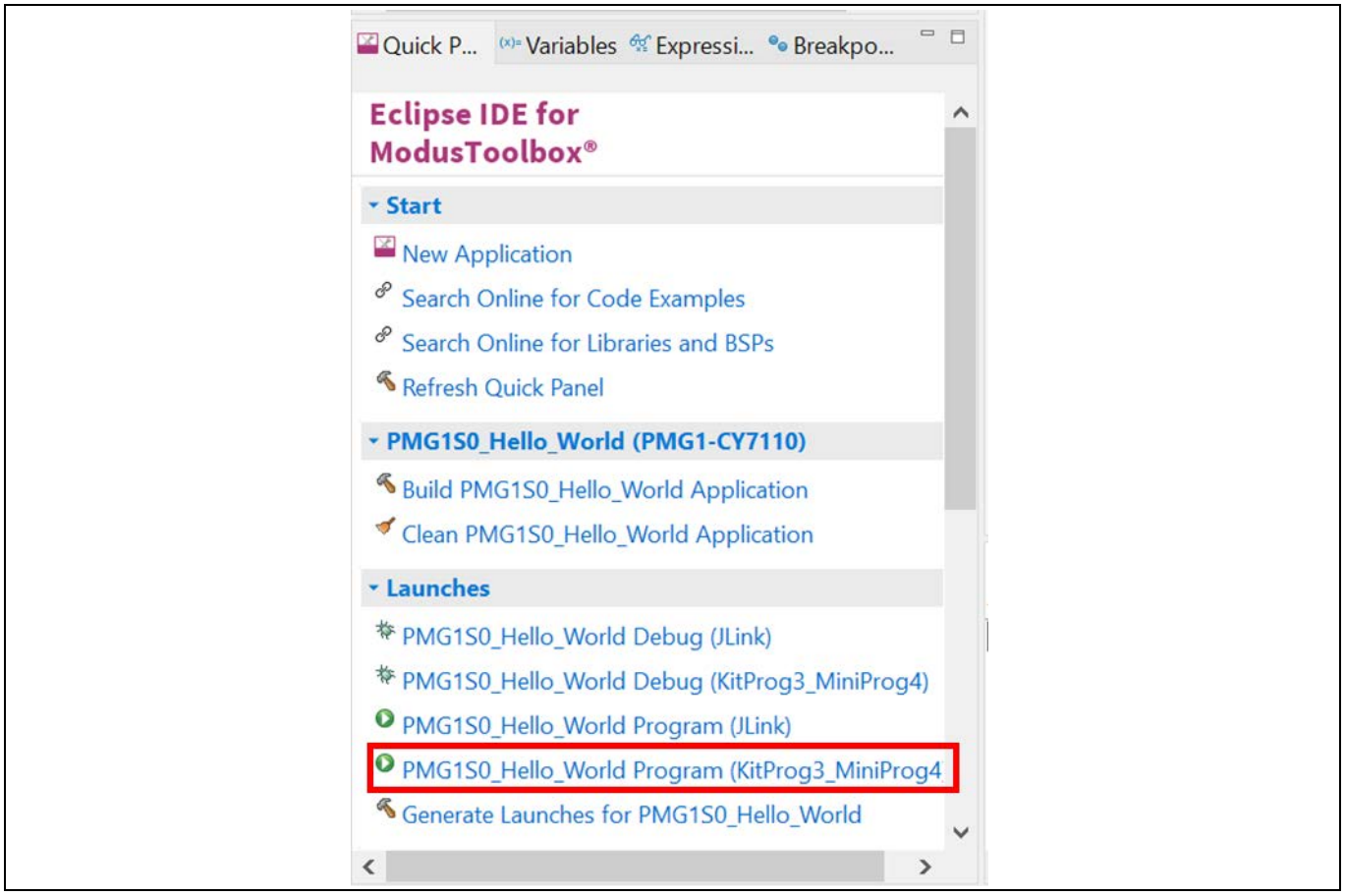


Figure 23　快速面板和调试选项

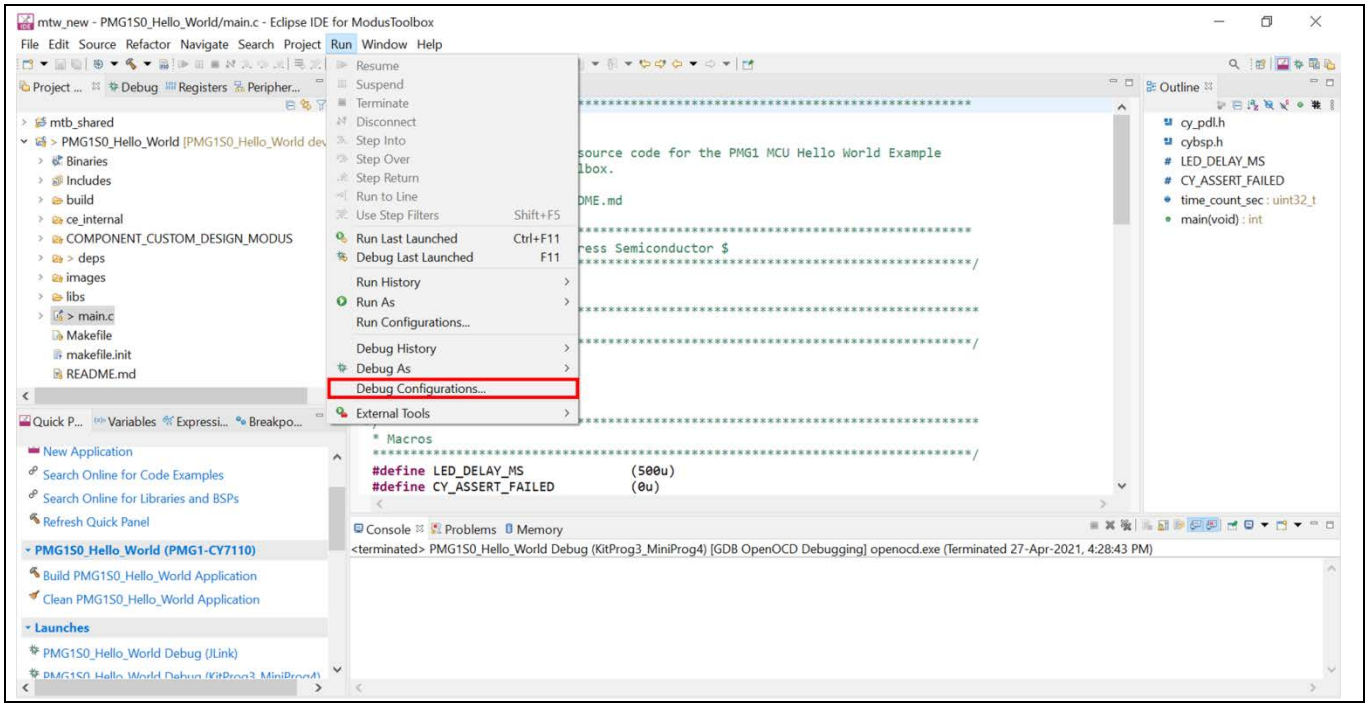


Figure 24　从 Run 菜单中选择调试配置

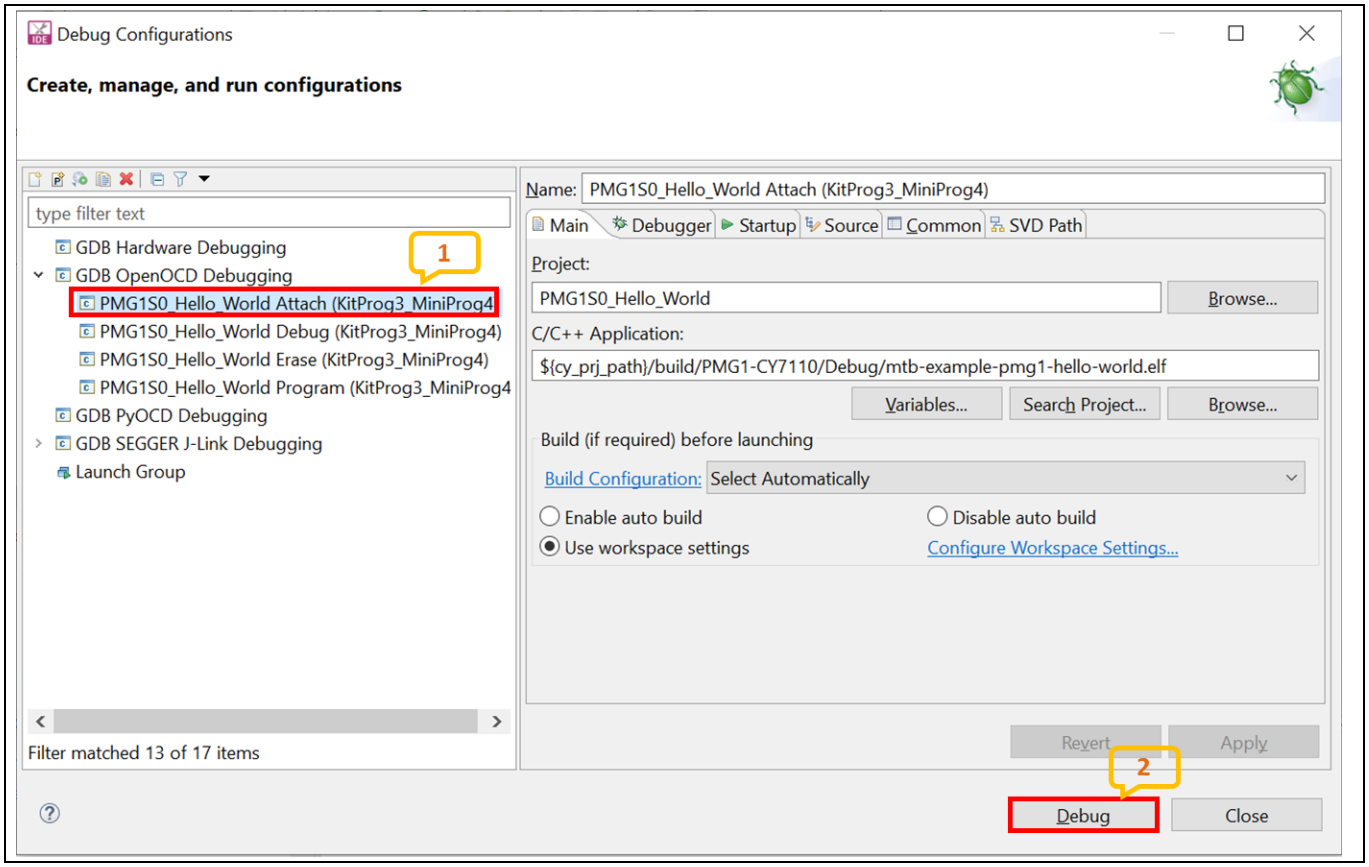


Figure 25　选择调试附加项并启动调试会话

成功完成编译、构建和调试启动后，ModusToolbox™软件将套件配置为调试模式，并且调试工具将出现在工具栏上。

3.4.7.2　通过 10 引脚 SWD 插头进行调试

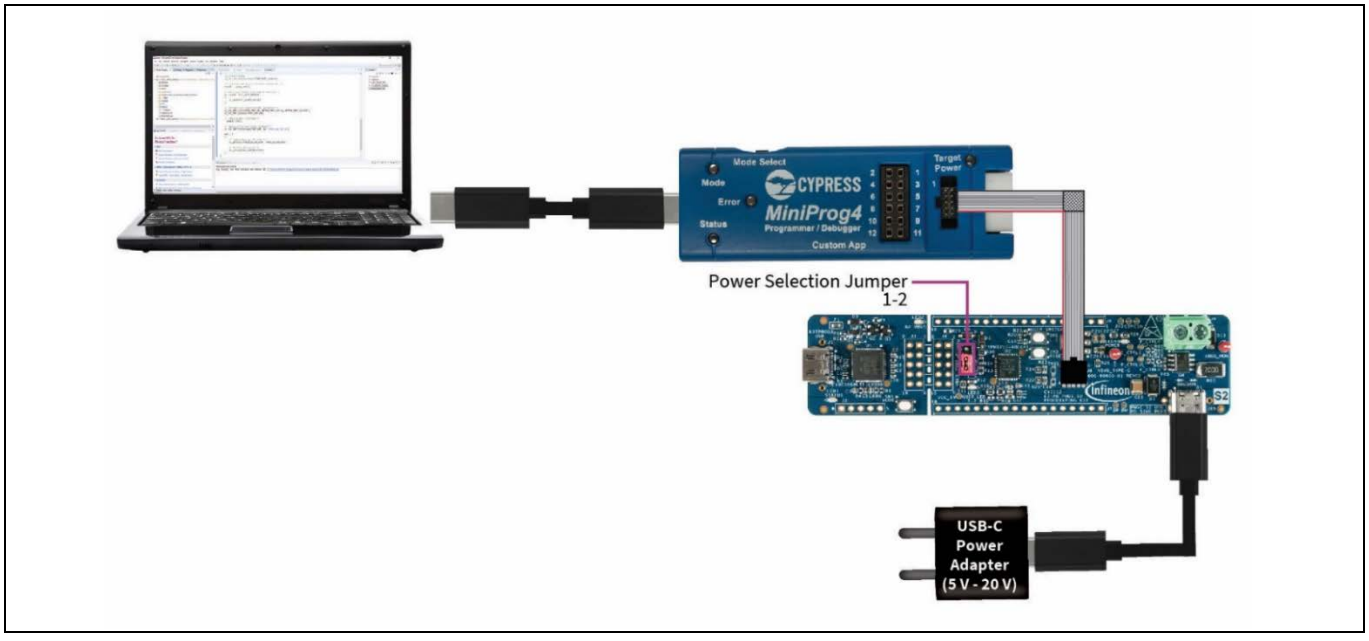


Figure 26　通过 10 引脚 SWD 插头进行调试

Figure 26 显示的是通过 10 引脚 SWD 连接器调试原型开发套件的设置。ModusToolbox™软件的 Eclipse IDE 的使用流程与 KitProg3 界面的使用流程相同。请按照**通过 KitProg3 接口进行调试**部分所介绍的第 5 到第 7 步骤进行操作。

3.5 添加中介软件

本节介绍了使用库管理器将中间件添加到有效应用程序中的各个步骤。对于“Hello World”示例项目，不需要添加任何额外中间件。

在 *main.c* 代码中完成中介软件的初始化。

1. 在 **Quick Panel** 中，点击 **Library Manager** 链接。
2. 在后续的对话框中，选择 **Libraries** 选项卡。
3. 在 **Board Utils or MCU Middleware** (电路板实用程序或 MCU 中介软件) 下方，选择并启用所需的中间件。
4. 点击 **Update** 按钮。

使用中介软件时所需要的文件被添加到 *mtb_shared* 文件夹以及 *deps* 文件夹下的.*mtb* 文件中。**Figure 27** 显示了添加中介软件的示例。指向 PDStack 中间件的.*mtb* 文件被添加到应用程序项目中。中间件内容也被下载并放置在名称为 **pdstack** 的相应文件夹中。

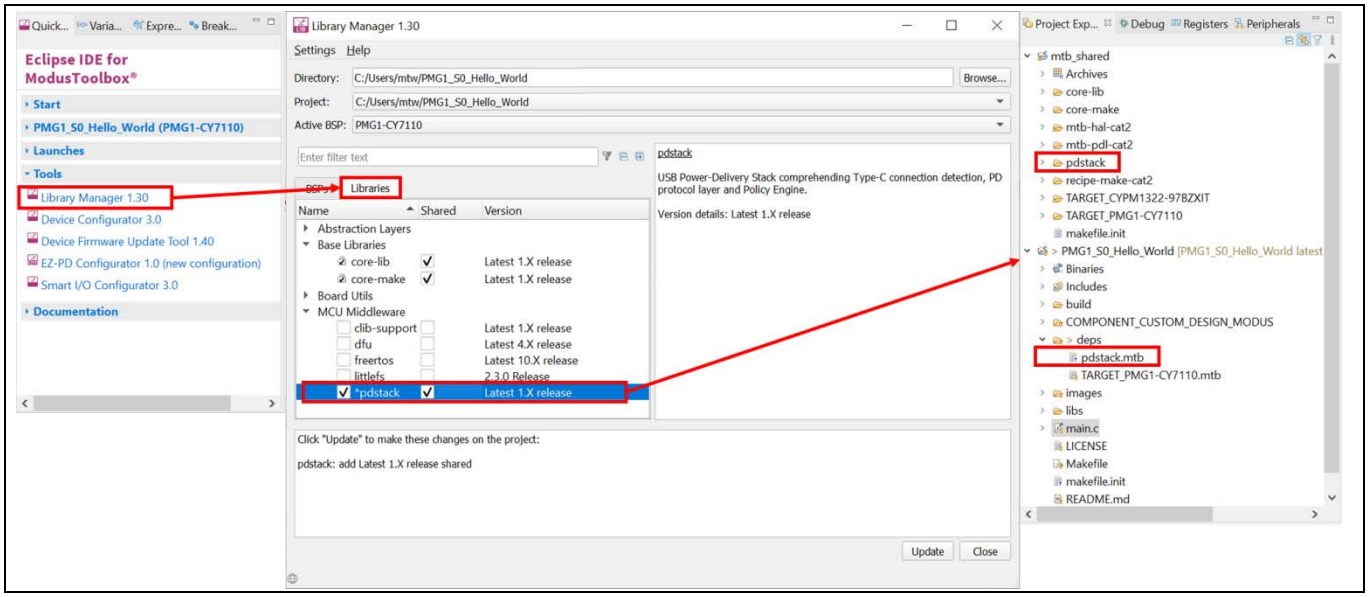


Figure 27 在 ModusToolbox™软件应用程序中添加 PDStack 中介软件

3.6 USB-PD 接收源示例

通过在 Select Application 对话框中选择“USBPD_Sink”模板应用程序，可以查看 USBPD 接收源的代码示例。

PMG1 MCU 器件支持 USB-PD 模块。该模块集成了 Type-C 终端、比较器和电力传输收发器，用于检测伙伴装置的附加与电源分配的协商。

该应用程序在 UFP (上行端口) — 接收源配置中使用 PDStack。PMG1 MCU 器件具有一个亏电电池 Rd 终端。即使没有给 PMG1 MCU 器件供电，该终端仍然能够确保与它连接的 USB-C 发送源/充电器能够检测到接收源的存在。

在大部分工作时间内，该应用程序尝试保持 PMG1 MCU 器件处于深度睡眠状态中。在该状态下，所有时钟都被禁用，并且只有有限数量的硬件模块被启用。对 USB-PD 模块中的中断进行配置，以检测器件处于睡眠模式时发生的所有变化，并将其唤醒以进一步进行处理。

USB-PD 模块中的过压 (OV) 比较器用于检测电源提供错误电压的情况，并在发生这种情况时自动关闭电源开关，以保护电路板上的其余组件。

应用创建成功后会打开 *README.md* 文件，该文件包含了模板应用实现的信息。

3.7 在各个 PMG1 MCU 之间导入项目

为某个 MCU 所创建的项目可以被导入到 PMG1 MCU 系列的其它 MCU 上。这里，我们假设新的 MCU 上具有正在导入的性能/功能/资源。要想在各个 PMG1 MCU 之间导入项目，请参见以下各步骤。

1. 从库管理器中打开快速链接。
2. 在库管理器中执行以下操作：
   a) 选择新的 BSP，以添加该项并取消选择当前的 BSP，从而将其删除。在 **Figure 28** 中，您可以看到 PMG1-CY7110 被取消选中，而 PMG1-CY7111 被选中。这意味着您正在将 PMG1 MCU“Hello World”代码示例导入到 PMG1-S1 MCU 中。

Note: *要将“Hello World”示例导入到自定义硬件中，请选择已创建的自定义 BSP。*

   b) 点击 **Update** 按钮。

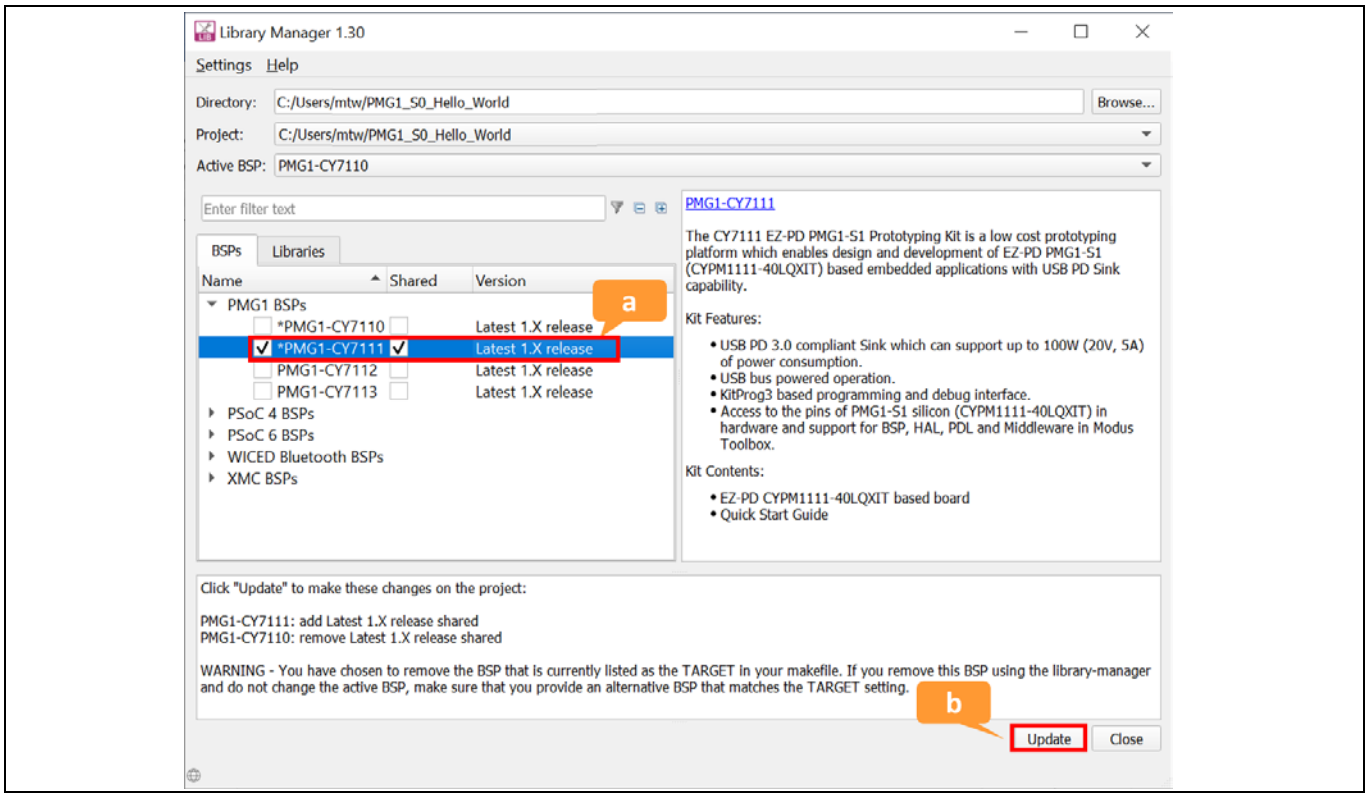


Figure 28 将项目导入到另一个 BSP 中

3. 在 Makefile 中，将目标名称更改为上一步骤中所选择的新 BSP 的名称。参见 **Figure 29**。

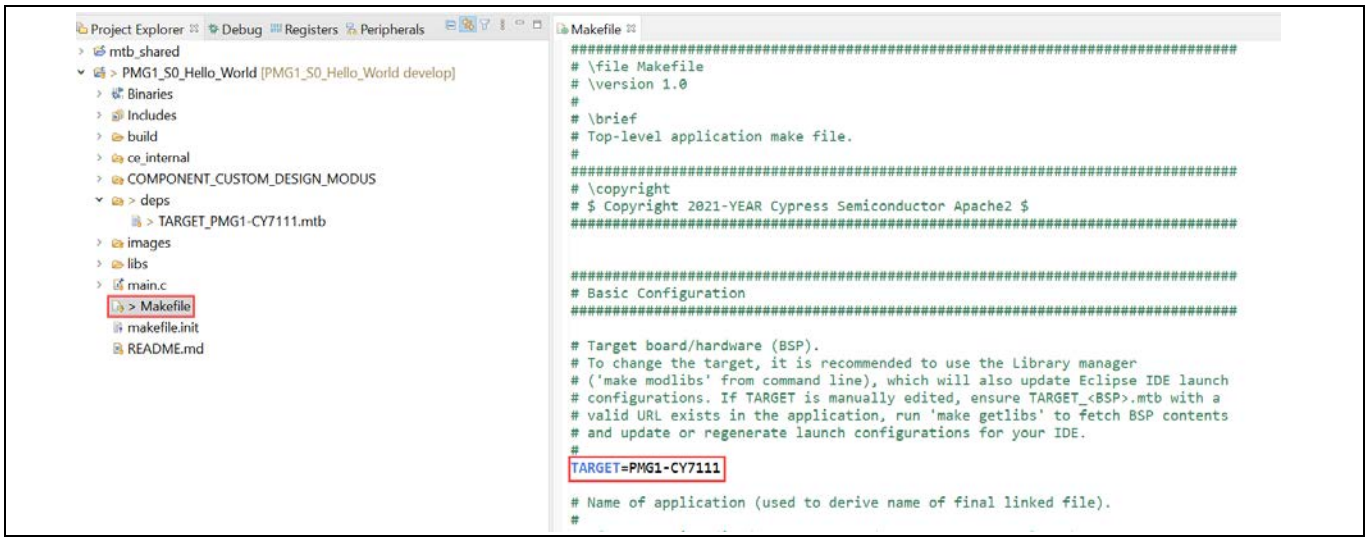


Figure 29　目标名称

4. 在快速面板中执行以下操作：
   a) 点击 Generate Launches for <项目名称>项。
   b) 点击 Refresh Quick Panel 项。

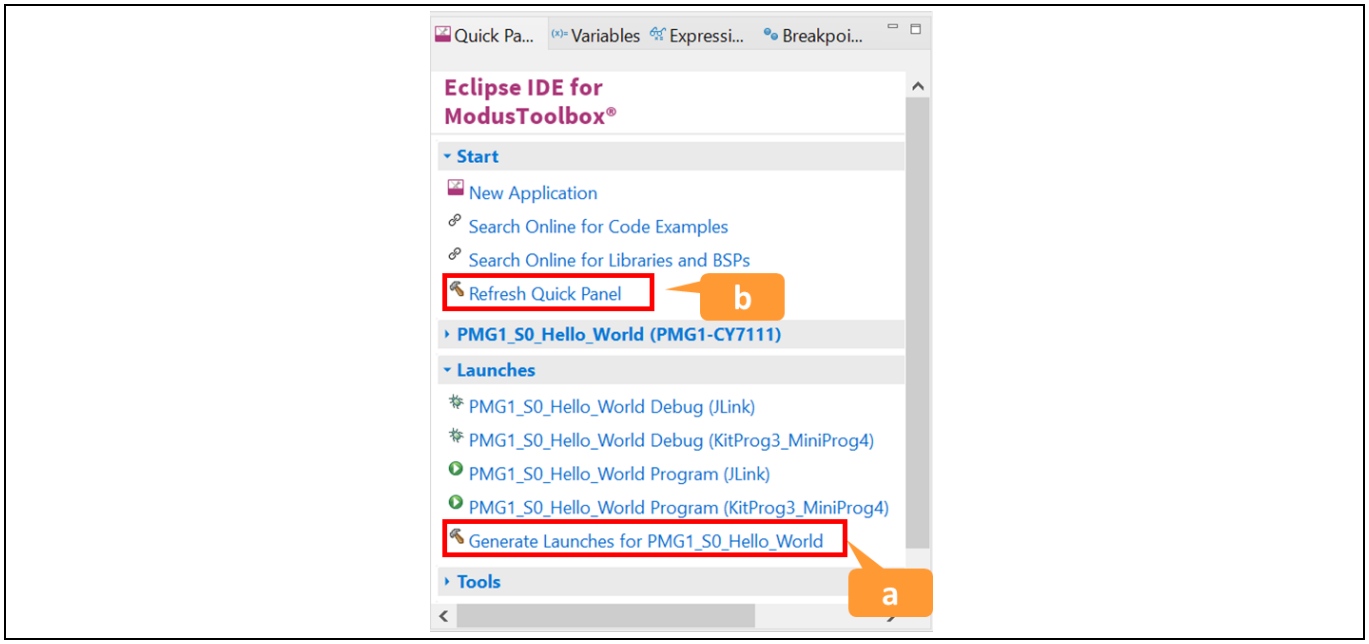


Figure 30　重新生成启动配置

目前，器件配置器已被更新为新的 BSP，如 **Figure 31** 所示。

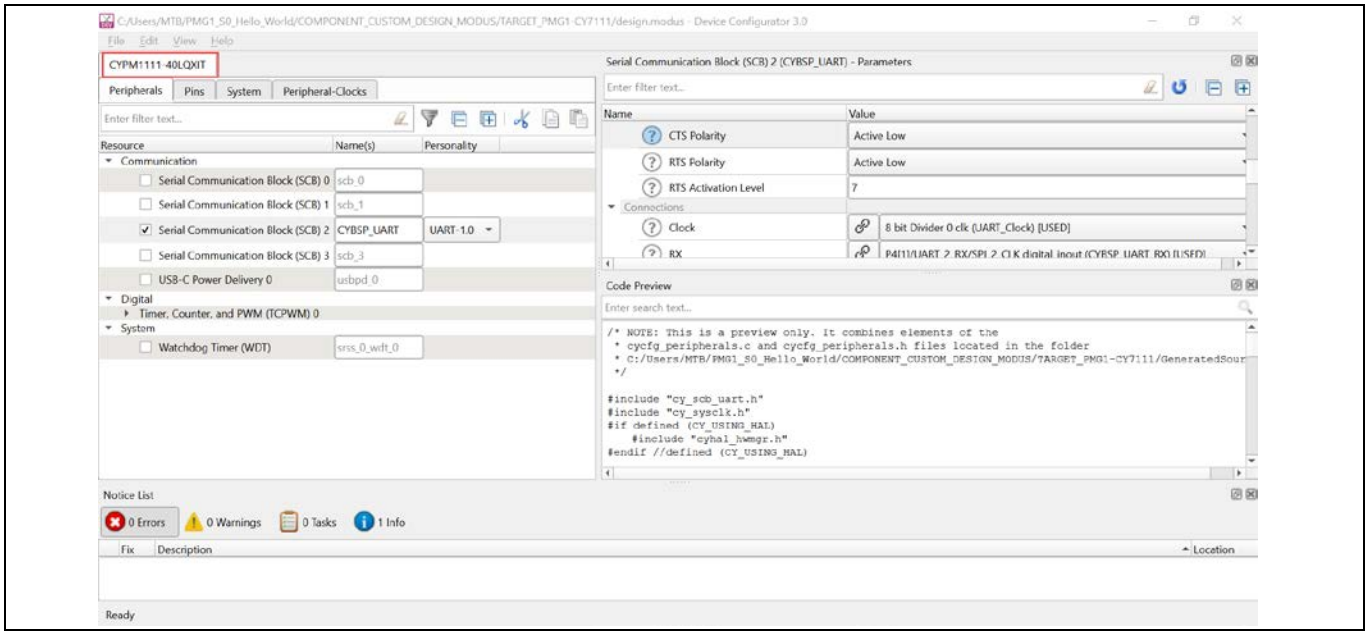


Figure 31　新 BSP 的器件配置器

5. 继续执行**应用程序调试流程中第四部分**以及后续部分中所讲述的其余开发步骤，从而构建、编程和测试项目。

4 总结

本应用笔记介绍了 PMG1 MCU 器件架构和相关的开发工具。PMG1 MCU 是可编程的嵌入式片上系统，它具有可配置的模拟和数字外设功能、存储器和 CM0/CM0+ CPU 系统。

相关文档

有关 PMG1 MCU 代码示例的完整和更新列表，请访问我们的 **GitHub 存储库**。更多有关 PMG1 MCU 的相关文档，请访问 **PMG1 MCU 产品网页**。

缩略语与首字母缩略词

缩略语	扩展内容
AC	苹果充电
AES	高级加密标准
AFC	适应性快速充电
BC	电池充电
CDM	充电器件放电模型
CRC	循环冗余检查
HBM	人体模型
OCP	过电流保护
OVP	过压保护
OVT	过压容限
PRNG	伪随机数生成
RCP	反向电流保护。仅在发送源配置中支持。
SCP	短路保护。仅在发送源配置中支持。
SHA	安全哈希 (hash) 算法
TRNG	真随机数生成
UVP	欠压保护

术语表

本部分列出了在 ModusToolbox™软件中操作 PMG1 MCU 器件系列时可能遇到的常用术语。

板级支持包 (BSP)： BSP 是包含了特定于电路板的驱动程序和其它功能的固件层。板级支持包是一个集库，它提供的固件 API 用于初始化电路板并对板级外设进行访问。

KitProg：KitProg 是一个具有 USB-I2C 和 USB-UART 桥接功能的板载编程器/调试器。大多数 PMG1 MCU 套件都集成了 KitProg。

MiniProg3 / **MiniProg4**：用于开发的编程硬件。可以用该硬件对自定义电路板上的 PMG1 MCU 器件或不支持内置编程器的 PMG1 MCU 套件进行编程。

性格：指的是资源对功能的可配置性。例如，可以将 SCB 资源配置为 UART、SPI 或 I2C 性能。

中介软件：中介软件是一组固件模块，可为应用程序提供特定功能。这样，可以为器件功能 (例如 PDStack) 提供高级软件接口。

ModusToolbox™软件：为开发人员提供的基于 Eclipse 的嵌入式设计平台。通过将驱动程序库、中间件和具有一流的感应能力的最低功耗、最灵活的 MCU 结合在一起，该平台可以提供单一、连贯且熟悉的设计体验。

外设驱动程序库：外设驱动程序库 (PDL) 简化了 PMG1 MCU 架构的软件开发。使用 PDL 时，不需要对寄存器使用和位结构有太多了解，从而简化了大量可用外设的软件开发。

修订记录

Document version	Date of release	Description of changes
**	2021-12-16	本文档版本号为 Rev**，译自英文版 002-32553 Rev*A。

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2021-12-16
Published by
Infineon Technologies AG
81726 Munich, Germany

© 2021 Infineon Technologies AG. All Rights Reserved.

Do you have a question about this document?
Go to www.cypress.com/support

Document reference
002-34376 Rev. **

重要提示

本文档所提供的任何信息**绝不应当**被视为针对任何条件或者品质而做出的保证（质量保证）。英飞凌对于本文档中所提及的任何事例、提示或者任何特定数值及/或任何关于产品应用方面的信息均在此明确声明其不承担任何保证或者责任，包括但不限于其不侵犯任何第三方知识产权的保证均在此排除。

此外，本文档所提供的任何信息均取决于客户履行本文档所载明的义务和客户遵守适用于客户产品以及与客户对于英飞凌产品的应用所相关的任何法律要求、规范和标准。
本文档所含的数据仅供经过专业技术培训的人员使用。客户自身的技术部门有义务对于产品是否适宜于其预期的应用和针对该等应用而言本文档中所提供的信息是否充分自行予以评估。

如需产品、技术、交付条款和条件以及价格等进一步信息，请向离您最近的英飞凌科技办公室接洽(**www.infineon.com**)。

警告事项

由于技术所需产品可能含有危险物质。如需了解该等物质的类型，请向离您最近的英飞凌科技办公室接洽。

除非由经英飞凌科技授权代表签署的书面文件中做出另行明确批准的情况外，英飞凌科技的产品不应当被用于任何一项一旦产品失效或者产品使用的后果可被合理地预料到可能导致人身伤害的任何应用领域。