MPLAB® X IDE 中的 CI/CD 向导用户指南

开发工具客户须知

重要：
开发工具手册如同所有其他文档一样具有时效性。我们不断改进工具和文档以满足客户的需求，因此实际使用中有些对话框和/或工具说明可能与本文档所述之内容有所不同。请访问我们的网站（www.microchip.com/）获取最新版本的 PDF 文档。

文档每页的底部均标有 DS 编号。DS 格式为 DS<文档编号><版本>，其中<文档编号>为 8 位数字，<版本>为大写字母。

有关最新信息，请访问 onlinedocs.microchip.com/查看您所使用的工具的帮助信息。

目录

开发工具客户须知.....1

1. CI/CD 向导.....3
   - 1.1. 关于 CI/CD.....3
   - 1.2. CI/CD 向导功能概述.....4
   - 1.3. Jenkins 入门.....4
   - 1.4. Jenkins 服务器入门.....15
   - 1.5. 如何通过 Unity 编写和运行单元测试.....17
   - 1.6. Dockerfile 入门.....27
   - 1.7. CICDW 命令行工具.....30

Microchip 网站.....32

产品变更通知服务.....32

客户支持.....32

Microchip 器件代码保护功能.....32

法律声明.....32

商标.....33

质量管理体系.....33

全球销售及服务网点.....34

1. CI/CD 向导

注：“MPLAB® X IDE User's Guide”（DS50002027）中也包含此内容。

*MPLAB X IDE CI/CD 向导*可帮助您针对具体的 MPLAB X IDE 项目设置自动编译和测试流水线。

CI/CD 要求

CI/CD 向导是 MPLAB X IDE 的一项功能，因此“Release Notes for MPLAB X IDE”中的相关要求对其同样适用。此外：

- 需要 MPLAB XC **工作站或网络服务器许可证**使能 CI/CD 向导生成脚本。
- 需要 MPLAB XC **网络服务器许可证**才能在生成的 Docker 容器中采用 PRO 优化级别编译项目。
- 如果使用**虚拟机网络许可证**，应确保该许可证与所有 Microchip **网络服务器许可证**位于同一文件夹中。
- 如果使用 Jenkins 流水线：
  - MISRA 检查（静态代码分析）和 MPLAB 代码覆盖（硬件测试）工具需要使用 MPLAB X IDE v6.00（或更高版本）和 MPLAB 分析工具套件**网络服务器许可证**才能在生成的 Docker 容器中正常执行。
  - 硬件测试需要使用带网络功能的工具，目前只有 MPLAB ICE 4 在线仿真器符合要求。MPLAB X IDE v6.00（或更高版本）支持该工具。
- 要使用 Jenkins 和/或 Docker 应用程序，需自行安装。有关使用 Jenkins 时的更多要求，请参见 1.3.1. 准备工作。
  注：MPLAB X IDE CI/CD 向导输出执行依赖第三方应用程序 Jenkins 和 Docker。有关这些工具的问题，请访问相应网站寻求支持。

CI/CD 标准使用

Jenkins-Docker 或 Docker 使用的标准操作系统是 Linux® OS，而 CI/CD 向导输出具体使用的是 Debian Linux。要使用其他操作系统，请参见 Jenkins 和 Docker 的文档。例如：

- JENKINS-60473：Docker 流水线：在 Windows 主机下运行 Linux 容器时的“绝对路径”错误

1.1 关于 CI/CD

CI/CD（或 CICD）是一种对软件进行连续集成以及连续交付或部署的过程。过去通常是在整个软件模块开发完毕后再尝试进行集成并最终交付产品，而现在则可以分阶段对软件进行连续集成以及连续交付或部署，从而确保在不同的生产阶段都能将产品的问题降至最少。

连续集成将使集成和测试活动提前进入开发周期。这有助于降低项目风险，从而提高产品成功概率。

要支持 CI/CD 流水线，可使用容器让整个团队共享一致的环境来编译和测试代码。通过开发指令文件来创建和运行容器（如 Dockerfile）以及运行容器内容（如 Jenkinsfile）并非易事。而这正是 CI/CD 向导的用武之地。

1.1.1 为什么将连续集成与编译和测试自动化相结合？

连续集成（Continuous Integration，CI）是一种频繁将代码更改提交给源代码资源库（如 git）的开发者实践。每次提交都会触发一个自动编译和测试周期。如果编译中断或测试失败，系统将发出通知。连续集成有助于从一开始便将高质量融入产品，不断与团队协调代码更改，最终降低与后期集成和后期验证反馈相关的风险。

连续集成开发者实践与**编译自动化**系统配合使用。该系统自动启动编译、测试、封装和部署等作业/过程。这有助于及早发现问题，从而使代码保持高质量。编译自动化工具有许多种。其中，Jenkins 是一款免费的开源软件，我们将以这款常用工具为例。如果使用其他系统，请参见由 CI/CD 向导生成的 Jenkinsfile 来获取相关信息。

1.1.2 Docker 和容器

编译自动化中的关键概念是**容器化**。执行编译作业时，应生成可重复且一致的结果，而与执行编译的位置和时间均无关。**Docker 容器**有助于实现这一点。使用 Docker 容器能够实现可重复性、灵活性和可扩展性。每次执行作业的环境均相同。Docker 提供 OS 级虚拟化，可将应用程序与基础架构分开。

有关如何安装 Docker 和使用 Dockerfile 的更多信息，请访问 www.docker.com。

1.2 CI/CD 向导功能概述

1. 创建定制的 **Dockerfile**。这些文件是用于生成 Docker 映像的配方。Docker 映像用于创建 Docker 容器。Docker 容器包含编译时所需的所有 MPLAB X IDE 项目设置，即 MPLAB X IDE 版本、MPLAB XC 工具链、XC 许可证处理、器件文件包（Device File Pack，DFP）和工具包。
2. 创建 **Jenkinsfile**。这些文件用于设置 Jenkins 流水线，该流水线说明如何使用 Docker 容器在 Jenkins 编译服务器上编译和测试 MPLAB X IDE 项目。
3. 处理编译服务器设置中的 MPLAB XC PRO 编译器许可证。
4. 使用 **Jenkins 服务器**创建并启动 Docker 容器。这样便可快速测试编译流水线或用作本地托管 Jenkins 服务器的基础。
5. 设置自动编译流水线以涵盖如下步骤：
   - MISRA 检查
   - 使用软件模拟器执行测试
   - 在硬件上执行测试
   - 收集代码覆盖数据
   - 使用 Doxygen 生成文档

1.3 Jenkins 入门

Jenkins 是一款开源自动化服务器，支持开发人员自动编译、测试和部署其软件。有关更多信息，请访问 Jenkins 官网。

本节介绍在 Jenkins 服务器上编译 MPLAB X IDE 项目所需的步骤。

1.3.1 准备工作

1.3.1.1 Jenkins 服务器

- 需要有 Jenkins 服务器。如果没有 Jenkins 服务器，可以按照 Jenkins 安装指南自行安装，或者使用 `cicdw jenkins-server` 命令创建本地测试服务器（见 1.4. Jenkins 服务器入门）。
- 需拥有用户帐户，以便在 Jenkins 服务器上配置新的编译作业。
- Jenkins 服务器需具有支持 Docker 的编译节点。如果没有，请参见 Jenkins 和 Docker。
- Jenkins 服务器必须安装以下插件：
  - Pipeline Aggregator View
  - Pipeline Utility Steps
  - Docker Pipeline
  - Workspace Cleanup
  - Warnings Next Generation

  注：可选编译步骤可能还需要其他插件，例如 MISRA 和软件模拟器。

1.3.1.2 源代码控制系统

- 项目源代码需托管在源代码控制系统（例如：Subversion、CVS 或 Git）上。
- Jenkins 服务器需支持源代码控制系统并能够访问项目资源库。

1.3.1.3 许可证服务器

- MPLAB XC 许可证服务器需配置为在给定的地址和端口上使用。有关更多信息，请参见 MPLAB XC 许可证服务器手册
- 要支持 MPLAB 分析工具套件（MISRA 检查和 MPLAB 代码覆盖）等其他许可选项，可能还需要其他许可证服务器。

1.3.1.4 测试工具

要使用软件（软件模拟器）或硬件（MPLAB ICE 4 在线仿真器）工具在容器内进行测试，必须考虑具体设置。

- 使用 Unity 执行单元测试并使用 `printf` 或 `putchar` 通过串行通信输出结果。对于硬件工具，目标还需要其他代码。有关 Unity 的更多信息，请参见 1.5. 如何通过 Unity 编写和运行单元测试。有关 MPLAB ICE 4 在线仿真器上的虚拟 COM 端口的信息，请参见用户指南的“USB CDC 虚拟 COM 端口”部分。
- 使用 Microchip 调试器（Microchip Debugger，MDB）命令行接口支持基本调试命令。对于硬件工具，使用 MDB 编程目标。有关 MDB 的更多信息，请参见 Microchip 调试器（MDB）用户指南。

1.3.2 步骤 1——在 MPLAB X IDE 中生成 Jenkinsfile

1. 在 MPLAB X IDE 中打开项目，然后导航到 *Tools > CI/CD Wizard*（工具 > CI/CD 向导）。
2. 选择 **Jenkins Pipeline files**（Jenkins 流水线文件）作为输出类型。
3. 按照向导完成各项操作，然后选择要在流水线中运行的工具。
4. 生成 *Jenkinsfile* 和 *Dockerfile* 后，验证它们是否已添加到项目基本文件夹中。此外，还可以根据所选的选项添加其他文件。

1.3.2.1 CI/CD 向导对话框 1

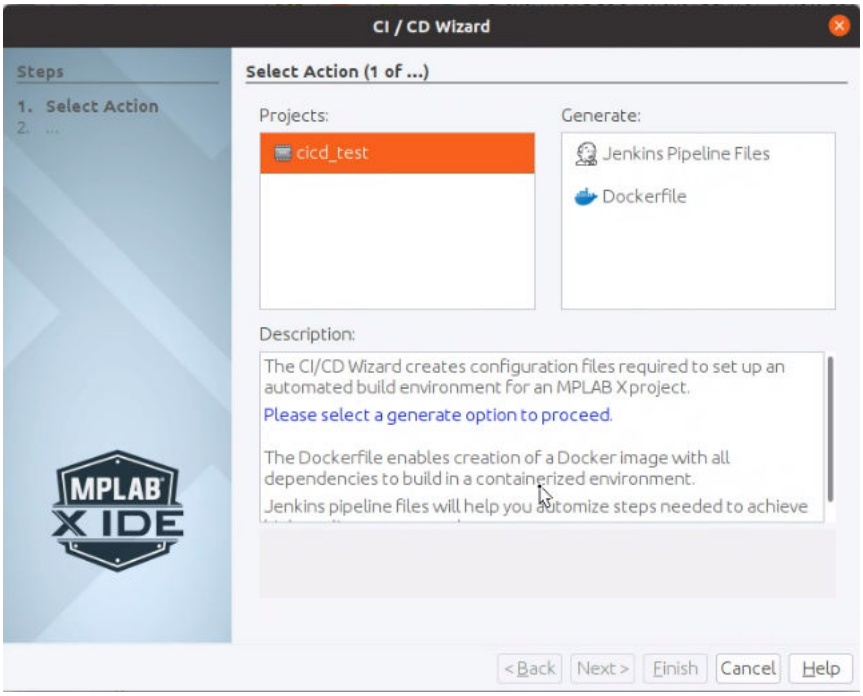
向导的第一个对话框中将显示如下说明（如果没有 MPLAB XC C 编译器许可证，将仅激活 **Cancel**（取消）按钮）：

要生成可在 CICD 流水线中使用的文件，需要拥有编译器许可证。

有关支持的许可证，请参见 1. CI/CD 向导。可以在 MySoftware 下为编译器购买许可证。单击 **Configure Licenses**（配置许可证）按钮可以转到 Change Licensing Type（更改许可类型）对话框来设置许可证。

如果拥有编译器许可证，将能够选择要生成的文件。选择 Jenkins 将同时创建 Jenkinsfile 和 Dockerfile。选择 Docker 将仅创建 Dockerfile。

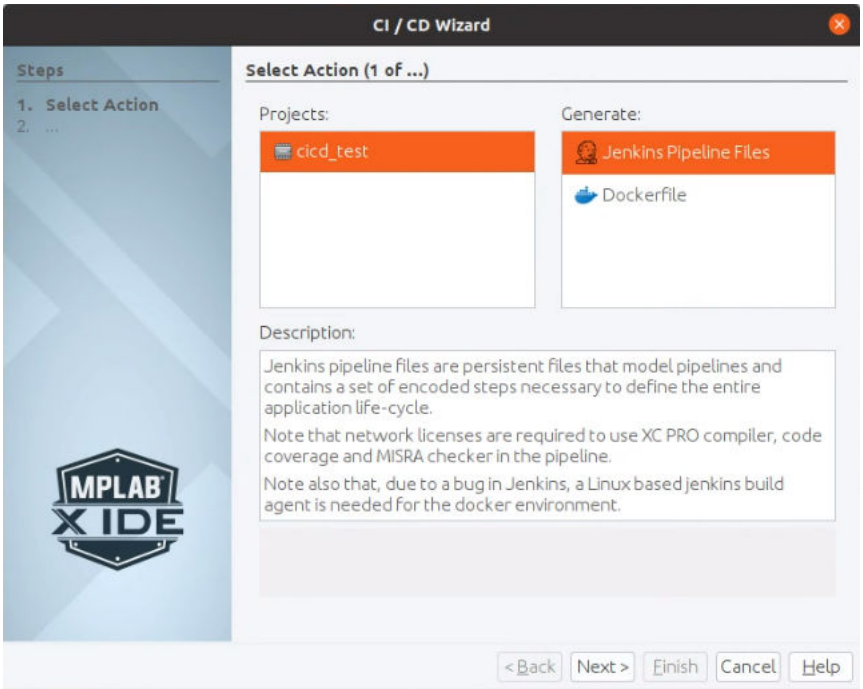
图 1-1. 不选择生成的文件



在本例中，选择“Jenkins Pipeline Files”（Jenkins 流水线文件）。阅读“Description”（说明）下有关该选择的文本。

单击 **Next**（下一步），继续完成对话框。

图 1-2. 选择生成的文件

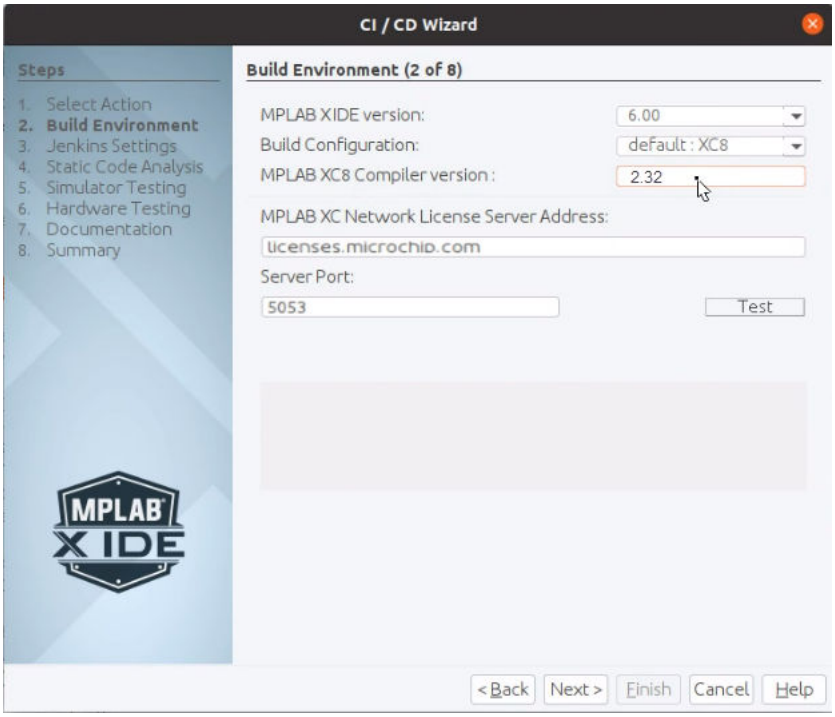


1.3.2.2 CI/CD 向导对话框 2

设置要在容器中使用的编译环境。如果拥有网络服务器许可证，则可以提供如何访问该许可证以进行编译的相关信息，甚至可以测试连接（见下图）。

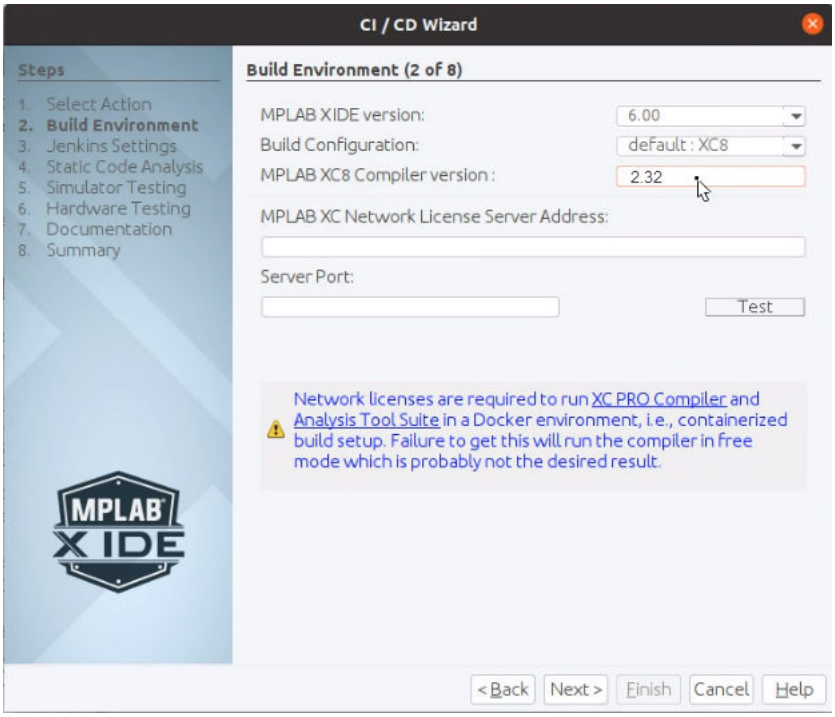
注：如果已使用 MPLAB X IDE *Tools > Licenses*（工具 > 许可证）设置网络服务器许可证，该对话框中将自动填充相应的信息。

图 1-3. 网络服务器许可证



否则将文本框留空（不过将出现一条说明）。

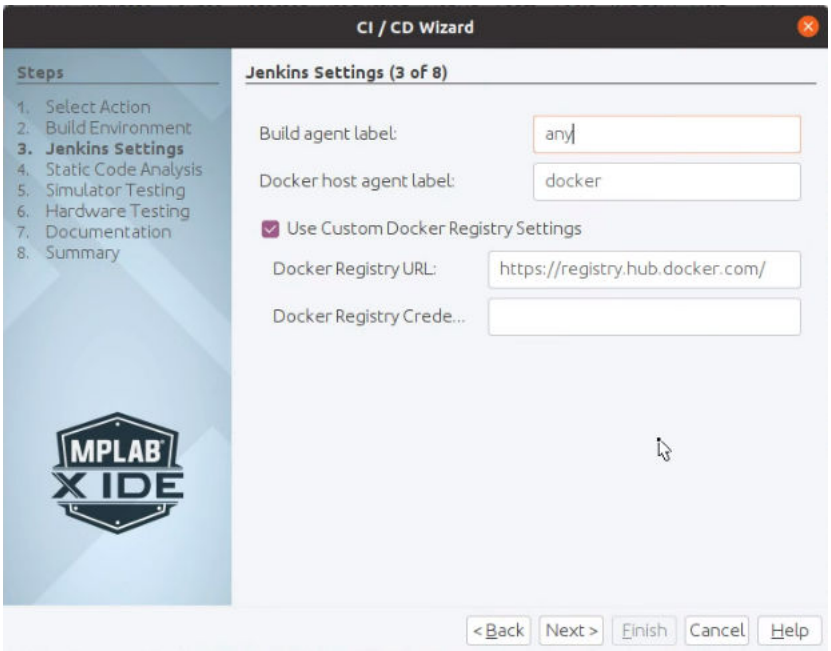
图 1-4. 工作站许可证



选项	说明
MPLAB IDE version（MPLAB IDE 版本）	从下拉列表中选择支持的版本。默认为最新版本。
Build Configuration（编译配置）	选择编译所需的项目配置。
MPLAB XC Compiler version（MPLAB XC 编译器版本）	选择编译所需的编译器版本。
MPLAB XC Network License Server Address（MPLAB XC 网络许可证服务器地址）	输入服务器的地址。例如：licenses.microchip.com。
Server Port（服务器端口）	输入服务器的端口号。例如：5053
信息性说明（视情况提供）	如果拥有工作站许可证，将显示有关购买网络服务器许可证的信息。

1.3.2.3 CI/CD 向导对话框 3

确定供 Docker 下载映像的位置。默认 Docker 信息将输入到顶部。要使用定制 docker 注册表，请输入以下信息。



选项	说明
Build agent label（编译代理标签）	默认设置为“any”（任何），但可以添加特定标签。请确保在 Jenkins 服务器上的 Configure System settings（配置系统设置）中为 Jenkins 代理添加相应的标签。
Docker host agent label（Docker 主机代理标签）	Docker 主机代理标签是位于 Jenkins 代理上的标签，支持 Docker 容器的编译和运行。请记住，可能需要遵循标准命名语法。关于这些命名约定，请参见 Jenkins 文档。
Use Custom Docker Registry Settings（使用定制 Docker 注册表设置）	使用自己的 docker 注册表来下载基本映像。更多相关信息，请参见如何使用自己的注册表。 基本映像将从 Docker Hub 下载。如有需要，可以指定定制基本映像的路径。
Docker Registry URL（Docker 注册表 URL）	注册表位置。
Docker Registry Credentials ID（Docker 注册表凭证 ID）	用于推送和拉取映像的凭证。有关详细信息，请参见使用 Jenkins——凭证。

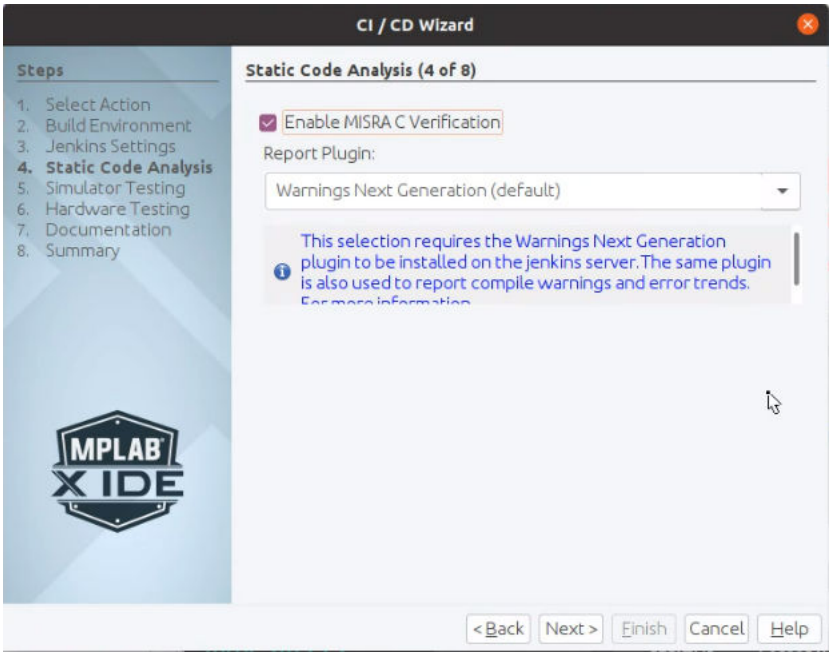
1.3.2.4 CI/CD 向导对话框 4

选中时可使能 MISRA C 验证。有关 MISRA 的更多信息，请访问 www.misra.org.uk/。

使用此类功能的要求如下：

- 选择任一报告都需要 Jenkins 插件。信息性说明将提供相关链接。
- MISRA C 验证仅适用于 MPLAB X IDE v6.00 及更高版本。如果当前版本不支持该功能，将显示信息性说明。
- 要在生成的 Docker 容器中执行工具，需要 MPLAB 分析工具套件网络服务器许可证。

图 1-5. Warnings Next Generation



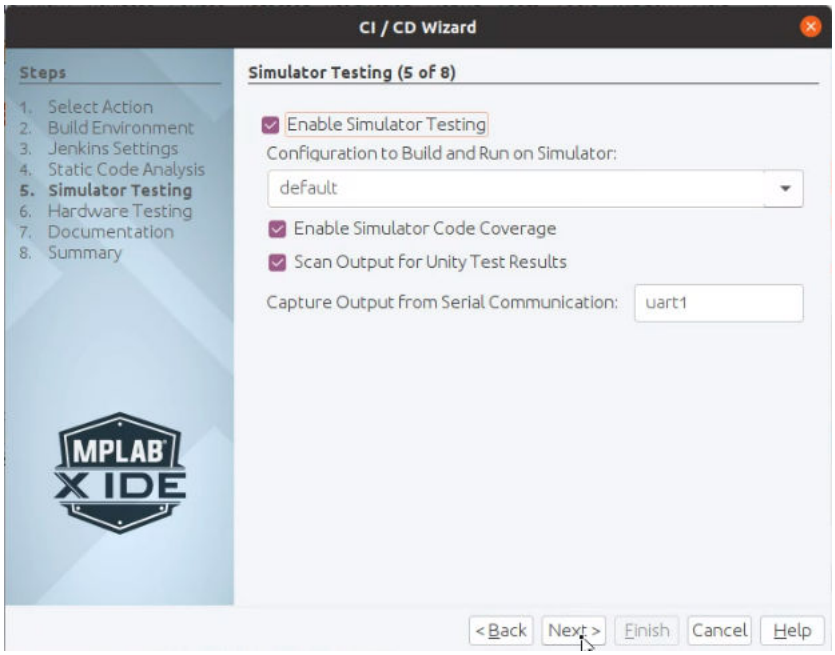
选项	说明
Enable MISRA-C Verification（使能 MISRA-C 验证）	选中时可使能 MISRA 验证。
Report Plugin（报告插件）	选择如下： • Warnings Next Generation • Cppcheck

1.3.2.5 CI/CD 向导对话框 5

选中时可使能软件模拟器用作测试映像，并可指定其他选项，以便用于运行单元测试。

Microchip 调试器（MDB）脚本 *mdb-simulator-script.txt* 将在软件模拟器编译步骤期间添加到项目中。

默认脚本将使软件模拟器运行 10 秒，并根据具体设置捕捉输出和覆盖信息。如需更改，则应手动修改脚本或使用其他选项重新生成。需要时根据测试需求定制输入文件。请参见 Microchip 调试器（MDB）用户指南。



选项	说明
Enable Simulator（使能软件模拟器）	使能软件模拟器作为调试工具。
Configuration to Build and Run on Simulator（通过软件模拟器编译和运行的配置）	选择使用软件模拟器的项目配置；即默认配置或软件模拟器专用配置。
Enable Simulator Code Coverage（使能软件模拟器代码覆盖）	使能软件模拟器内置的代码覆盖功能。 **注：**如果使能覆盖报告，必须在 Jenkins 服务器上安装代码覆盖 API 插件。
Scan Output for Unity Test Results（扫描输出以获取 Unity 测试结果）	在按照项目配置编译 Unity 测试运行程序时使能，编译作业随后将基于生成的输出创建报告。有关如何编写 Unity 测试的更多信息，请参见 1.5. 如何通过 Unity 编写和运行单元测试。
Capture Output from Serial Communication（通过串行通信捕捉输出）	该选项应与 *Scan output for Unity test results* 选项配合使用。软件模拟器需要知道从哪个串行通信模块实例捕捉输出。默认值为 *uart1*，但具体取决于使用的器件以及测试设置的配置方式。

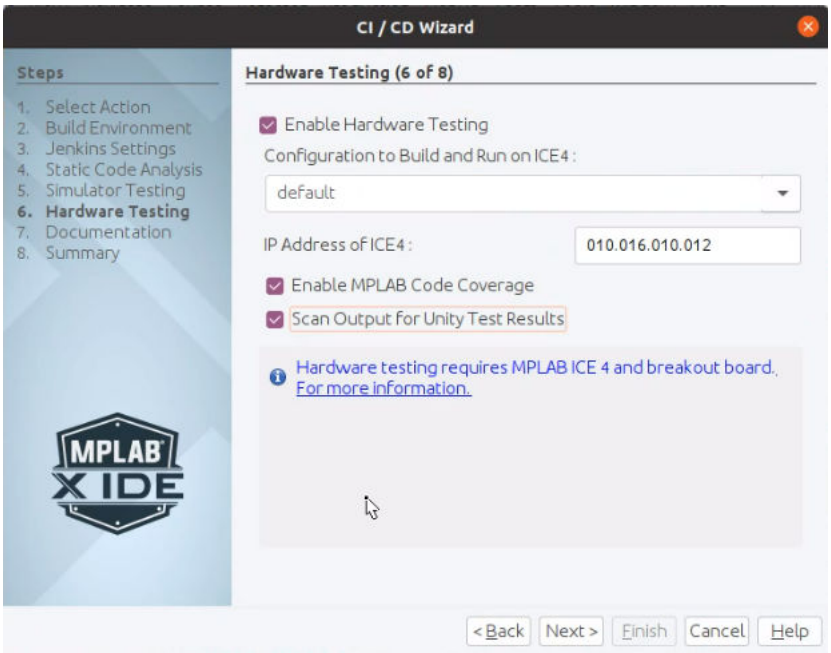
1.3.2.6 CI/CD 向导对话框 6

选中时可使能硬件工具用作测试映像，并可指定其他选项，以便用于运行单元测试。

注：MPLAB ICE 4 在线仿真器是目前惟一一款带网络功能的硬件工具，并且只有 MPLAB X IDE v6.00（或更高版本）支持该工具。

Microchip 调试器（MDB）脚本 *mdb-hardware-script.txt* 将在硬件编译步骤期间添加到项目中。

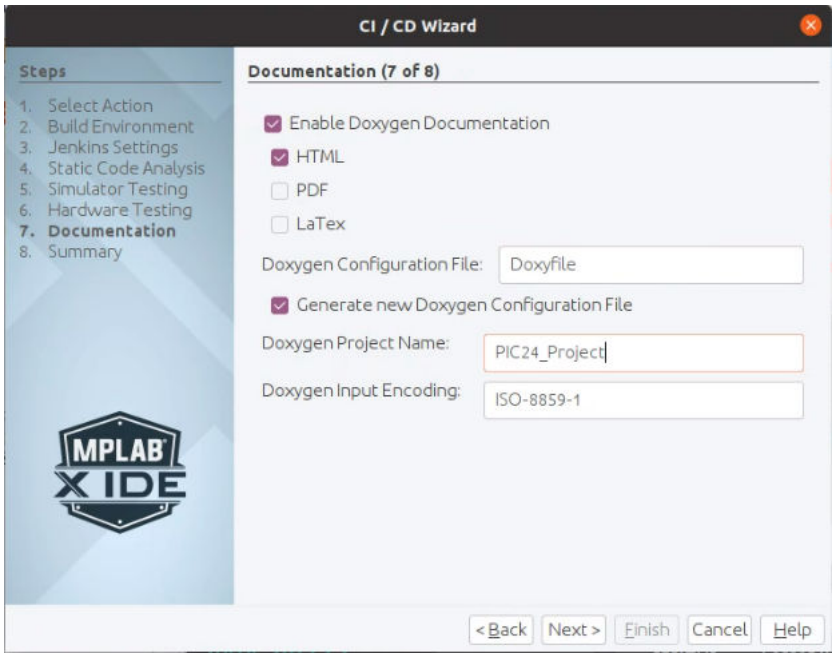
默认脚本将运行 10 秒，并根据具体设置捕捉输出和覆盖信息。如需更改，则应手动修改脚本或使用其他选项重新生成。需要时根据测试需求定制输入文件。请参见 Microchip 调试器（MDB）用户指南。



选项	说明
Enable Hardware Testing（使能硬件测试）	使能仿真器作为测试工具。
Configuration to Build and Run on ICE4（通过 ICE4 编译和运行的配置）	选择使用仿真器的项目配置；即默认配置或硬件专用配置。
IP Address of ICE4（ICE4 的 IP 地址）	输入要使用的仿真器的 IP 地址。如果仿真器的 IP 地址未知，可打开 *Tools > Manage Network Tools*（工具 > 管理网络工具），然后扫描网络工具。随后将列出仿真器及其 IP 地址。
Enable MPLAB Code Coverage（使能 MPLAB 代码覆盖）	使能 MPLAB 代码覆盖功能。要在生成的 Docker 容器中执行工具，需要 MPLAB 分析工具套件网络服务器许可证。 **注：**如果使能覆盖报告，必须在 Jenkins 服务器上安装代码覆盖 API 插件。
Scan Output for Unity Test Results	在按照项目配置编译 Unity 测试运行程序时使能，编译作业随后将基于生成的输出创建报告。有关如何编写 Unity 测试的更多信息，请参见 1.5. 如何通过 Unity 编写和运行单元测试。

1.3.2.7 CI/CD 向导对话框 7

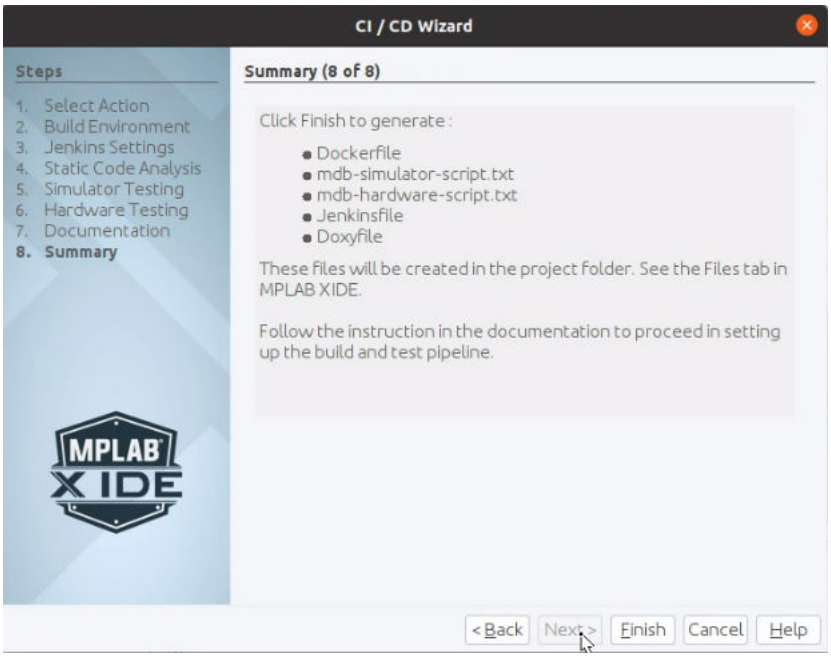
选中时可使能 Doxygen 作为文档工具，并可指定输出。有关 Doxygen 的更多信息，请参见 GitHub 上的 Doxygen。



选项	说明
Enable Doxygen Documentation（使能 Doxygen 文档）	使能文档工具。
Documentation Format（文档格式）	选择一种或多种输出格式。
Doxygen Configuration File（Doxygen 配置文件）	指定配置文件。
Generate new Doxygen Configuration File（生成新的 Doxygen 配置文件）	选中时可创建新文件或覆盖现有文件。取消选中时可保留现有文件。
Doxygen Project Name（Doxygen 项目名称）	指定项目名称。
Doxygen Input Encoding（Doxygen 输入编码）	指定文档编码。

1.3.2.8 CICD 向导对话框 8

在 Summary（汇总）屏幕上查看预期输出。如需更改，应单击 **Back**（上一步）。



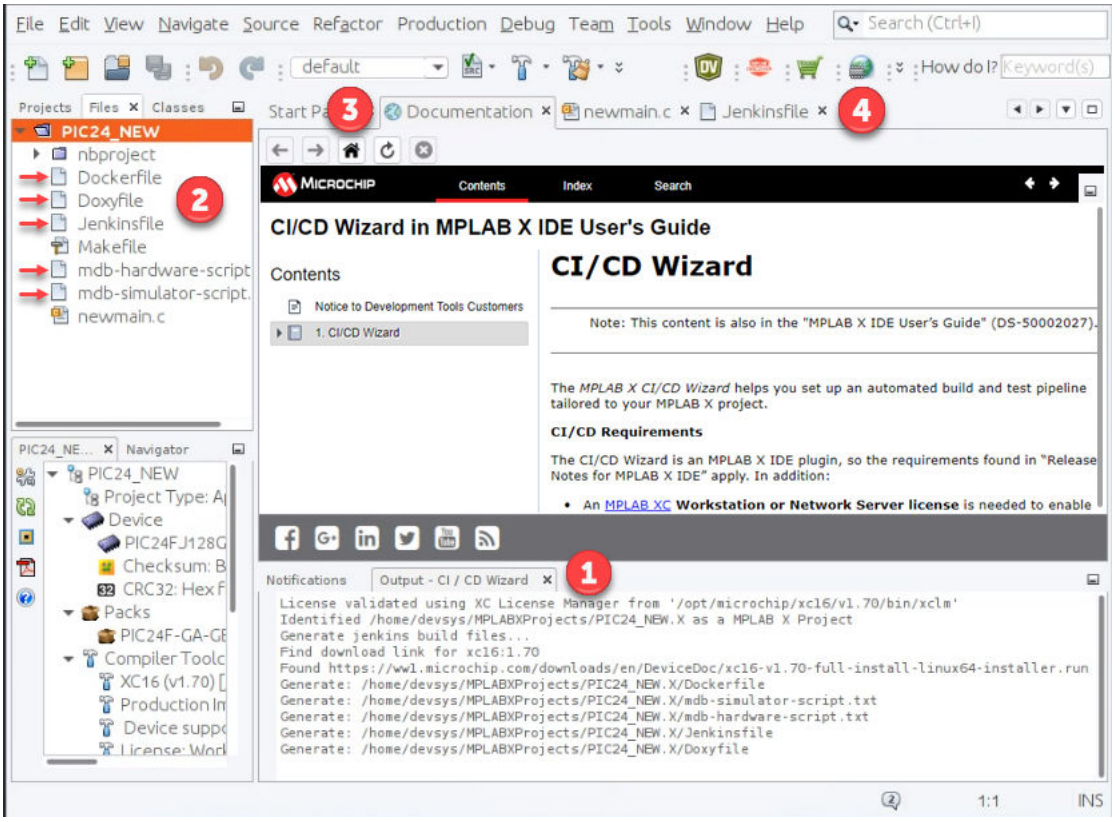
单击 **Finish**（完成）完成设置。CI/CD 向导随后将根据设置信息生成文件。

1.3.2.9 CI/CD 向导输出

在 MPLAB X IDE 中，可以看到以下内容：

1. Output（输出）窗口中正在生成的文件。
2. 生成后添加到项目文件夹中的文件。
3. WebHelp（在单独窗口中）。
4. Jenkinsfile（在单独窗口中）。

图 1-6. MPLAB X IDE 中的文件生成



所生成文件的详细信息：

Jenkinsfile

包含 Jenkins 流水线定义的文本文件。有关更多信息，请参见使用 Jenkinsfile。

Dockerfile

用于编译容器的定制文本文件。有关更多信息，请参见 Dockerfile 参考。

mdb-simulator-script.txt

只有在助手的步骤 5 中选择了软件模拟器时才会生成该文件。该文件用于设置软件模拟器在 Microchip 调试器（MDB，即无 GUI 形式的 MPLAB X IDE）中的用途。可以根据测试需求定制该文件。有关更多信息，请参见 Microchip 调试器（MDB）用户指南。

mdb-hardware-script.txt

只有在助手的步骤 6 中选择了 MPLAB ICE 4 在线仿真器时才会生成该文件。该文件用于设置仿真器在 Microchip 调试器（MDB，即无 GUI 形式的 MPLAB X IDE）中的用途。可以根据测试需求定制该文件。有关更多信息，请参见 Microchip 调试器（MDB）用户指南。

Doxygen

只有在助手的步骤 7 中选择了文档时才会生成该文件。

Doxygen 是一种基于带注释的源代码生成文档的标准工具。有关配置文件格式的详细信息，请参见配置。

1.3.3 步骤 2——将 Jenkinsfiles 添加到资源库中

使用源代码控制系统，将生成的文件添加到项目资源库中。Jenkins 会将该资源库中的项目添加到创建的容器中。

1.3.4 步骤 3——在 Jenkins 服务器上配置编译作业

本节概述了如何使用生成的文件在 Jenkins 服务器上配置新的编译作业。有关更多详细信息，请参见在 Jenkins.io 上使用 Jenkinsfile。

概述：

1. 登录 Jenkins 服务器 Web 界面，然后选择 **New Item**（新建项）。
2. 输入编译作业名称，然后选择 **Pipeline**（流水线）作为项目类型。
3. 在新的编译作业配置中，转到 **Pipeline** 部分进行配置：
   a. 选择 **Pipeline script from SCM**（来自 SCM 的流水线脚本），然后配置 Jenkins 服务器签出项目所需的源代码控制管理（Source Control Management，SCM）信息。其中可能包括*资源库 url*、*凭证*和*分支说明符*。
   b. 指定 *Jenkinsfile* 作为 **Script Path**（脚本路径）。（需要与所生成 Jenkinsfile 的名称相匹配）。
   c. 保存新的编译作业配置并启动新的编译作业（**Build Now**（立即编译））。

结果应是正常工作的编译流水线，该流水线首先设置 docker 编译环境，然后编译和分析 MPLAB X IDE 项目。

1.3.5 建议的后续步骤

Jenkinsfile 定义在 docker 映像内运行的编译步骤。docker 映像由生成的 *Dockerfile* 定义，可提供稳定的 Linux 环境，该环境中包含所需的 MPLAB XC 工具链、器件支持和 MPLAB X IDE 二进制文件。

可以按照 Jenkinsfile 文档修改生成的 Jenkinsfile。

1.3.5.1 门控

当编译作业稳定后，便可执行门控提交过程：每次提交都将自动接受测试，然后再判断是被拒绝还是合并到主分支中。此过程可确保即使有多个开发人员在同时开发一个项目，主分支也依然可以保持稳定。

门控编译过程可以采用不同的方式进行设置，具体取决于使用的源代码管理基础架构。这里给出了有关如何使用 Atlassian Bitbucket 集成的示例。

1.4 Jenkins 服务器入门

Jenkins 是一款开源自动化服务器，支持开发人员自动编译、测试和部署其软件。有关更多信息，请访问 Jenkins 官网。

CI/CD 向导提供了一种在 Docker 容器中运行自动配置的 Jenkins 服务器实例的解决方案。Jenkins 服务器可配置为编译来自本地文件夹或 git 的现有 MPLAB X IDE 项目。该服务器既可作为进一步配置和定制的起点，也可作为针对由向导生成的编译流水线的本地测试平台。

1.4.1 准备工作

Docker

Docker 是一组平台即服务产品，使用 OS 级虚拟化以软件包（称为容器）形式交付软件。各容器之间彼此隔离并绑定各自的软件、库和配置文件。

- 有关更多信息，请访问 docker.com
- Docker 入门

1.4.2 创建并启动 Jenkins 服务器

为了创建并启动 Jenkins 服务器，将使用 CI/CD 向导的命令行接口 `cicdw`。有关该工具的更多信息，请参见 1.7. CICDW 命令行工具。

1.4.2.1 创建“普通”Jenkins 服务器

要创建未预配置任何编译作业的普通 Jenkins 服务器，请输入以下命令

```
cicdw jenkins-server create
```

1.4.2.2 创建具有基于 Git 的编译作业的 Jenkins 服务器

要创建预配置了基于 git 资源库进行编译的编译作业的 Jenkins 服务器，请输入以下命令：

```
cicdw jenkins-server create --with-git-job origin
```

该命令将提取与本地 git 资源库的 git 远程命名“origin”相关的 git 服务器信息。为此，项目路径（默认为当前文件夹）必须指向包含 Jenkinsfile 并签入远程 git 资源库的有效 MPLAB X IDE 文件夹。

注：如果尚无与项目相关的 Jenkinsfile，请使用 CI/CD 向导来生成文件。请参见 1.3. Jenkins 入门。

1.4.2.3 创建具有本地编译作业的 Jenkins 服务器

要创建预配置了基于本地文件夹进行编译的编译作业的 Jenkins 服务器，请输入以下命令：

```
cicdw jenkins-server create --with-local-job
```

触发后，编译作业将在计算机的本地文件夹中查找 Jenkinsfile。启动服务器时，使用[`path`]设置提供本地文件夹路径（并非生产服务器模式）。

注：如果尚无与项目相关的 Jenkinsfile，请使用 CI/CD 向导来生成文件。请参见 1.3. Jenkins 入门。

1.4.2.4 启动 Jenkins 服务器

Jenkins 服务器创建完毕后，可通过输入以下命令进行启动：

```
cicdw jenkins-server start
```

请注意，配置了本地编译作业的服务器将使用分配给启动命令的项目路径（默认为当前文件夹）作为其本地文件夹。

启动后，可通过浏览器访问 Jenkins 服务器。

默认位置为 http://localhost:8080/，后续可使用`--server-name` 和`--server-port` 参数进行更改。

默认用户使用凭证 **mplabcicd/mplabcicd** 创建，后续可使用`--admin-user` 和`--admin-password` 参数进行更改。

1.4.3 凭证

当与 git 资源库等外部资源交互时，通常将需要多种凭证来验证和授权编译系统。Jenkins 凭证由凭证管理器处理，可在创建新服务器时提供。

```
cicdw jenkins-server create --credentials [TYPE::]ID::USERNAME[::PASSWORD][::KEYFILE] ...
```

凭证分为两种类型：`usr_pwd` 或 `sshkey`。

`usr_pwd` 为默认类型，因此参数`--credentials my_id::my_username::my_password` 将解析为 `TYPE=usr_pwd`, `ID=my_id`, `USERNAME=my_username`, `PASSWORD=my_password`。

对于 `sshkey` 类型，如果只提供四个元素，则第四个元素将解析为 keyfile 的路径。如果提供五个元素，则第四个元素将解析为 keyfile 的密码，第五个元素将解析为 keyfile 的路径。

`--credentials` 标志后可列出一个或多个凭证参数。

1.4.3.1 示例

```
cicdw jenkins-server create --credentials httpuser::theuser::secret123
```

或

```
cicdw jenkins-server create --credentials usr_pwd::httpuser::theuser::secret123
```

将创建使用用户名 `theuser` 和密码 `secret123` 作为凭证 id `httpuser` 的用户名/密码凭证。

```
cicdw jenkins-server create --credentials sshkey::sshuser::theuser::path/to/.ssh/key
```

将创建使用用户名 `theuser` 和来自文件夹 `path/to/.ssh/` 的 keyfile `key` 作为凭证 id `sshuser` 的 ssh 凭证。

```
cicdw jenkins-server create --credentials
sshkey::sshuser::theuser::keyfilepwd::path/to/.ssh/key
```

与前一个示例基本相同，惟一的区别是使用密码 `keyfilepwd` 来解锁 keyfile。

1.4.3.2 为编译作业使用凭证

基于 git 的编译作业通常需要使用凭证。要使用由`--credentials` 参数创建的凭证，应添加`--git-job-credentials-id` 参数。简称为`--gjci`。

```
cicdw jenkins-server create --credentials cred_id::user::pwd --git-job-credentials-id cred_id
```

请注意，用户名/密码验证不适用于基于 ssh 的任何远程 url。

- ssh://git@some-git-server.com/projects/project.git
- ssh://user@some-git-server.com/~/projects/project.git
- git://some-git-server.com/projects/project.git
- git@some-git-server.com:projects/project.git

1.4.4 编辑 Jenkins 服务器编译文件

默认情况下，`cicdw jenkins-server create` 命令会将其配置文件写入临时文件夹中。通过包含`--build-path` 参数，可更改为更永久的位置。添加`--only-generate` 标志可以只生成编译文件，而不创建 docker 映像。

```
cicdw jenkins-server create --only-generate --build-path my/build/files/location
```

该命令将在相应的文件夹中生成三个文件。

- Dockerfile——有关如何为 Jenkins 服务器编译 docker 映像的说明
- plugins.txt——Jenkins 服务器中要预安装的插件的列表
- casc.yaml——Jenkins 服务器的配置说明

有关如何编辑这些文件的信息，请参见官方 JCasC 文档。

要使用编辑后的文件，应将`--from-build-files` 标志添加到 `cicdw jenkins-server create` 命令中。请注意，必须先删除之前创建的所有服务器才能创建新服务器。

```
cicdw jenkins-server remove
cicdw jenkins-server create --build-path my/build/files/location --from-build-files
```

1.5 如何通过 Unity 编写和运行单元测试

Unity 是一款由 MIT 许可的框架，可用于轻松编写单元测试。有关官方入门指南，请参见：Unity——入门。

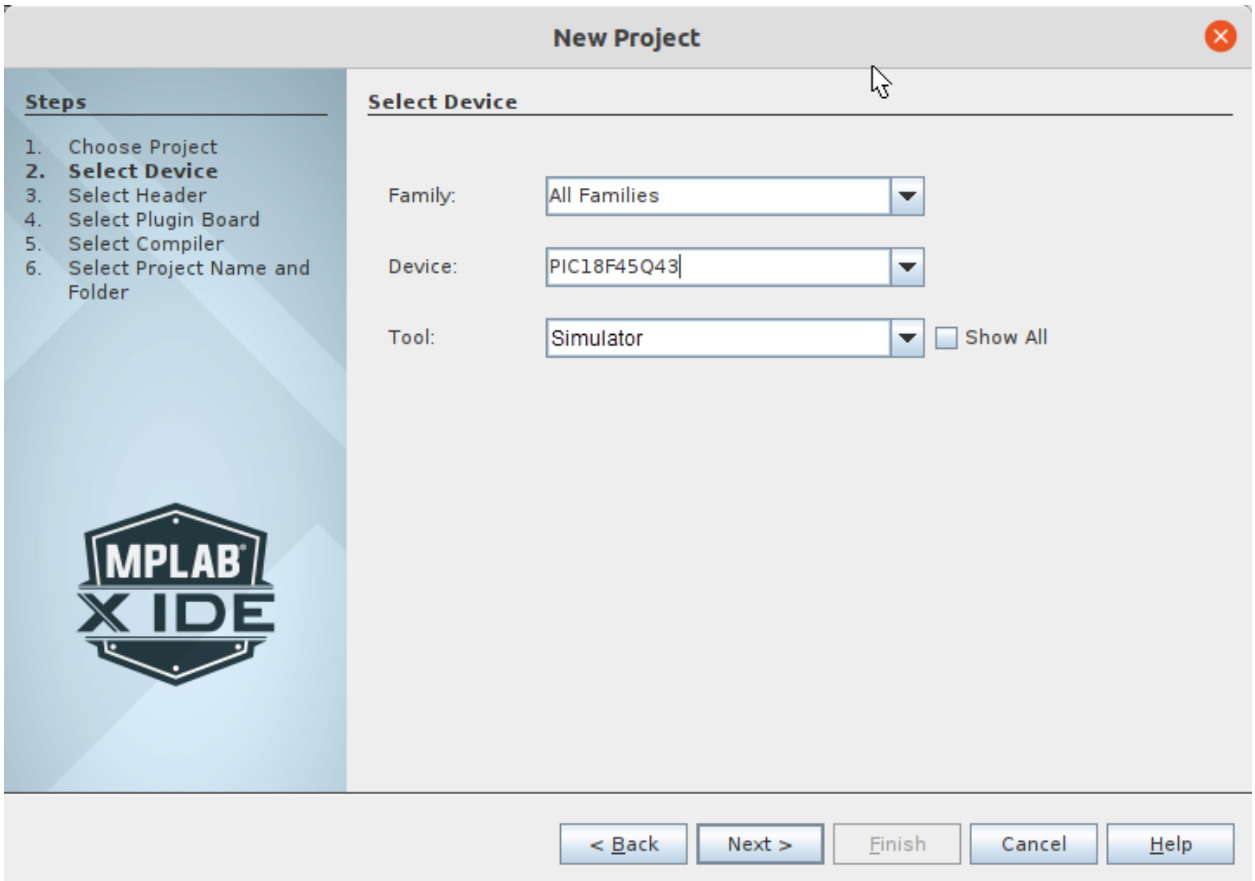
有关如何在 MPLAB X IDE 环境中使用该产品的示例，请参见以下章节。

1.5.1 通过软件模拟器运行单元测试

下面举例说明将 Unity 与 MPLAB X 软件模拟器配合使用的操作步骤。

1.5.1.1 步骤 1. 创建新项目

在 MPLAB X IDE 中创建新的独立项目（ ），其中包括选择器件和工具链。

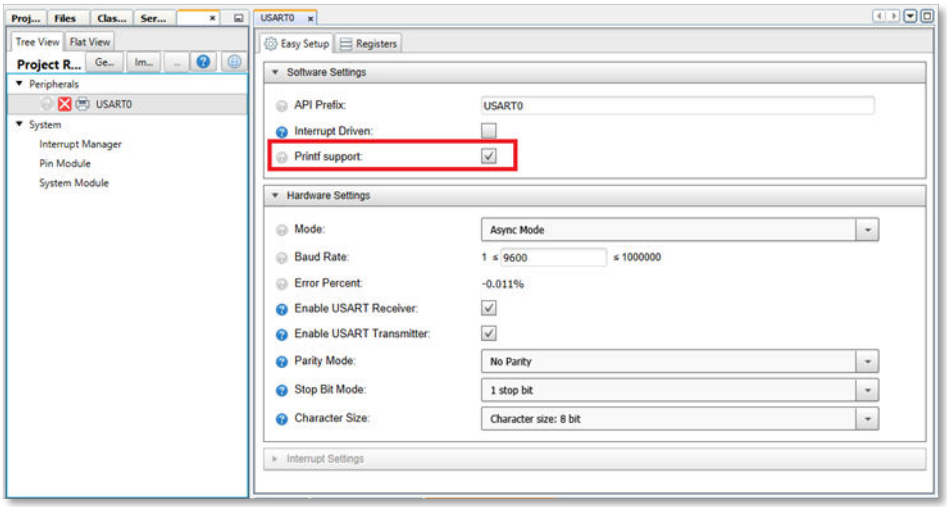


1.5.1.2 步骤 2. 配置串行输出

Unity 需要支持 *printf* 才能输出测试结果。这可以通过多种方法来实现。

其中一种方法是安装并使用 MPLAB X IDE 插件 *MPLAB 代码配置器*（MPLAB Code Configurator，MCC）。安装后，打开 MCC 并从 *Device Resources*（器件资源）视图添加 USART（或 UART）。确保在配置中使能 *Printf* 支持。

以下示例显示了在 MCC 中为 *ATtiny817* 器件配置的 USART，其中使能了 *Printf* 支持。



在 MCC 中配置完 USART（或 UART）后，记得单击“Generate”（生成）并验证配置的文件是否添加到项目中。

1.5.1.3 步骤 3. 通过 MDB 脚本和软件模拟器验证串行输出

应先验证串行输出和 `printf` 是否按预期工作，然后才能继续使用 Unity。

步骤 3.A——在 main.c 中测试 printf

设置将因所选器件和 `printf` 设置方式而异。

以下 `main.c` 示例假设器件为 ATtiny817，并且配置的 USART 由 MCC 在 `mcc_generated_files` 子文件夹中生成：

```
#define F_CPU 1000000UL

#include "mcc_generated_files/mcc.h"
#include <util/delay.h>

int main(void)
{
    // Initialize drivers from MCC
    SYSTEM_Initialize();
    _delay_ms(1000);

    printf("Hello world!\n");
}
```

步骤 3.B——创建软件模拟器 MDB 脚本

创建通过软件模拟器本地运行程序并将串行通信记录到本地文件中的 MDB 脚本。

下面给出了一个使用 ATTiny817 的 `mdb.script` 示例：

注：需要修改 elf 文件的路径。

```
# Set tool and device
device ATtiny817
hwtool SIM

# Write serial communication from usart0 to file.
set usart0io.uartioenabled true
set usart0io.output file
set usart0io.outputfile ./sercom_output.txt

# Program and run your binary file
program ./dist/[INSERT_YOUR_BINARY_FILE_PATH_HERE].elf

run
# Wait 10 seconds before exiting
wait 10000
halt
quit
```

步骤 3.C——运行 MDB 脚本并验证输出

确保项目已编译，然后打开命令行，导航到项目。使用 mdb 脚本作为参数运行 `mdb.sh` 或 `mdb.bat`。

此处给出了一个示例：

```
me@mypc:/project_with_unittests.X$ "/opt/microchip/mplabx/6.00/mplab_platform/bin/mdb.bat"
mdb.sh

device ATtiny817

hwtool SIM

Resetting SFRs
Resetting peripherals

set usart0io.uartioenabled true
set usart0io.output file
set usart0io.outputfile ./sercom_output.txt
```

```
program /project_with_unittests.X/dist/default/production/
project_with_unittests.X.production.elf
Programming target...
Resetting SFRs
Resetting peripherals
Program succeeded.

run
Running

wait 10000
halt
Simulator halted
quit
```

如果现在打开 sercom_output.txt，应输出以下内容：

```
Hello world!
```

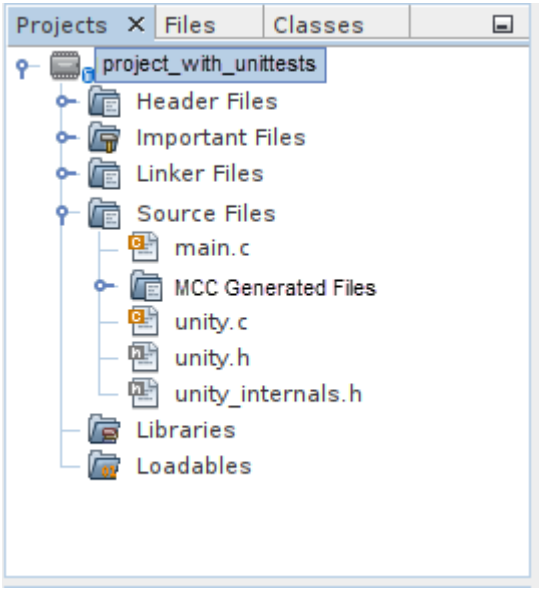
1.5.1.4 步骤 4. 添加 Unity 测试框架

现在已有正常工作的串行输出，可以添加单元测试。

需要手动下载 Unity 测试框架并将其添加到项目中。所需文件可从 https://github.com/ThrowTheSwitch/Unity/tree/master/src 下载。

下载所需文件并将其添加到项目中。

项目现在应如下所示：



1.5.1.5 步骤 5. 编写 Testrunner

重新编写 main.c 并向其中添加测试。有关如何编写更高级测试的信息，请参见：https://github.com/ThrowTheSwitch/Unity/blob/master/docs/UnityGettingStartedGuide.md。

例如：

```
#define F_CPU 1000000UL

#include "mcc_generated_files/mcc.h"
#include <util/delay.h>

#include "unity.h"

void test_function1() {
    // Simple demo of working test
    TEST_ASSERT_TRUE(1);
```

```
}

void test_function2() {
    // Simple demo of failing test
    TEST_ASSERT_FALSE(1);
}

int run_unit_tests(void)
{
    UnityBegin("main.c");
    RUN_TEST(test_function1);
    RUN_TEST(test_function2);
    UnityEnd();
    return 0;
}

int main(void)
{
    // Initialize drivers from MCC
    SYSTEM_Initialize();
    _delay_ms(1000);

    run_unit_tests();
}

void setUp (void) {}
void tearDown (void) {}
```

1.5.1.6 步骤 6. 通过软件模拟器运行测试

重新运行已创建的 mdb 脚本（如步骤 3.C 所述）。

此时的输出应如下：

```
main.c:21:test_function1: PASS
main.c:15:test_function2: FAIL: Expected FALSE Was TRUE

-----------------------
2  Tests 1  Failures 0 Ignored
 FAIL
```

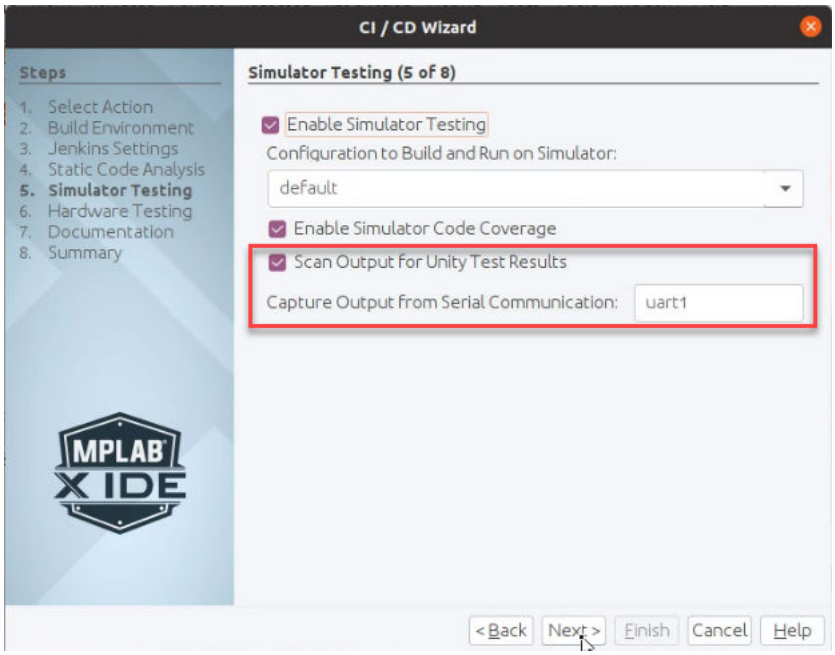
示例 mdb 脚本仅等待 10 秒便退出。 如果测试需要更多时间，可以在 `mdb.script` 中的 `wait` 命令中延长时间。

1.5.1.7 步骤 7. 通过 CI/CD 向导使用软件模拟器

完成可在本地正常工作的稳定设置后，应运行 CI/CD 向导并集成测试。

生成 Jenkinsfile 时，在 CI/CD 向导中使能软件模拟器测试步骤。

使能“Scan Output for Unity Test Results”并填入可供捕捉输出的串行通信实例的名称：

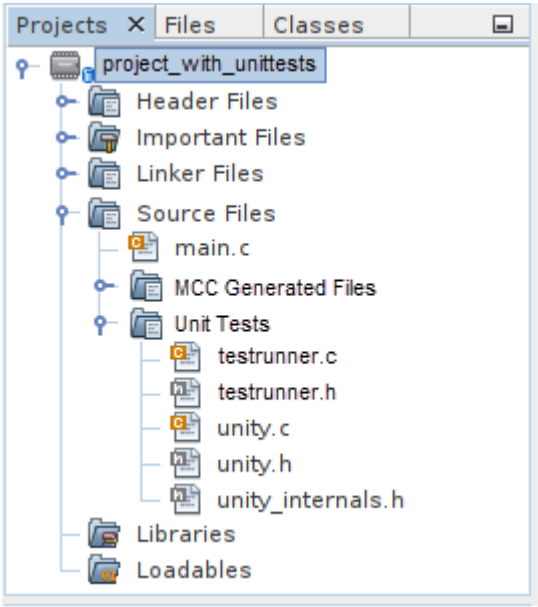


向导将生成其自己的 mdb 脚本并在 Jeknins 编译期间使用。测试结果将通过由 Jenkins 在编译后提供的输出解析。

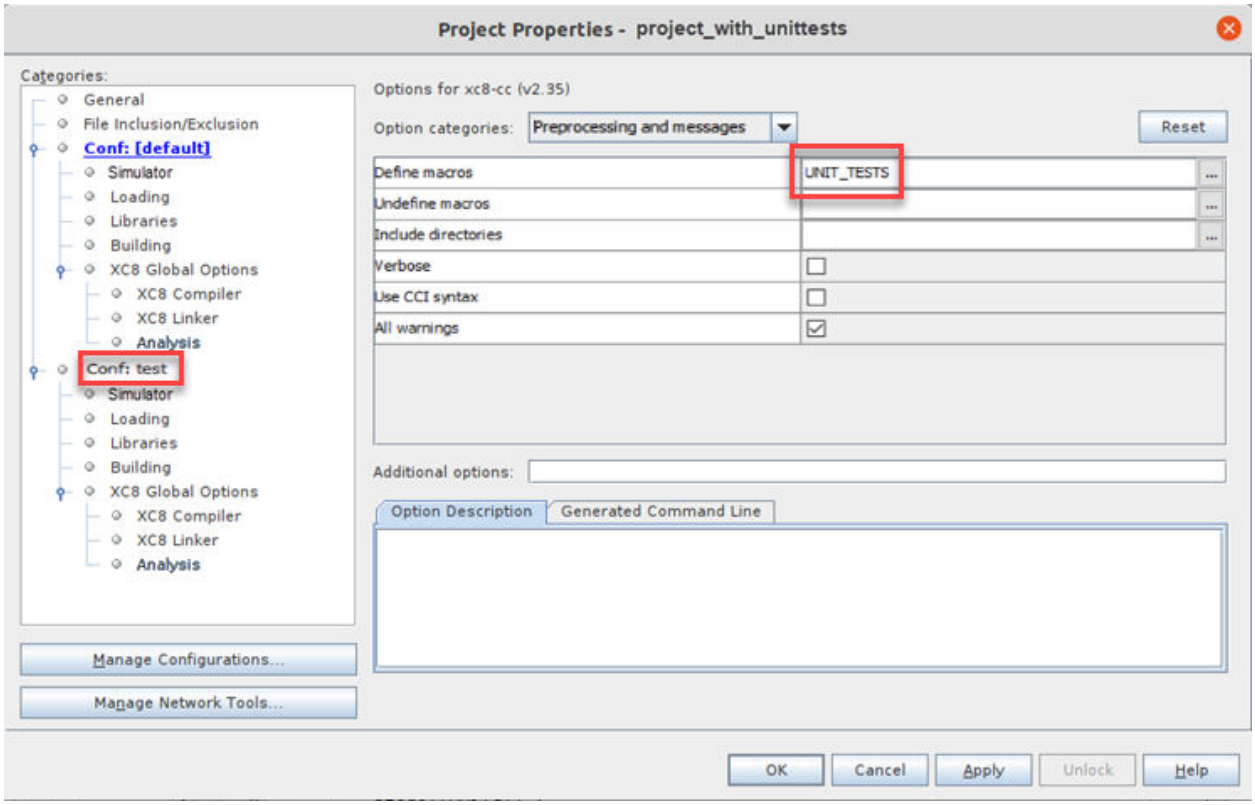
1.5.1.8 可选步骤

将测试代码和生产代码分开

为了更好地组织项目，应将单元支持文件与 testrunner 移至单独的子文件夹：



还可将单独的测试配置添加到项目中：



由于已完成定义 UNIT_TESTS 的测试配置（见上图）并且已将测试运行移至单独的子文件夹，因此可以将 `main.c` 修改为如下形式：

```
#include "mcc_generated_files/mcc.h"
#include <util/delay.h>

#ifdef UNIT_TESTS
#include "unit_tests/testrunner.h"
#else
int do_normal_stuff();
#endif

int main(void)
{
    // Initialize drivers from MCC
    SYSTEM_Initialize();
    _delay_ms(1000);

#ifdef UNIT_TESTS
    printf("Starting unit tests\n");
    run_unit_tests();
#else
    printf("Starting normal execution\n");
    return do_normal_stuff();
#endif
}

int do_normal_stuff() {
    return 0;
}
```

这样便可获得一个常见项目，只是一个配置优化生产编译，一个测试配置运行测试。

添加代码覆盖

如果要使能代码覆盖，可以在 mdb.script 文件中的 program 命令之前添加以下几行代码：

```
# Generate simulator coverage data and write it to a file
set codecoverage.enabled Enabled_Reset_on_run
set codecoverage.enableoutputtofile true
set codecoverage.outputtofile simulator_coverage.txt
```

这样可以在运行 mdb 脚本时将覆盖数据写入 simulator_coverage.txt。

1.5.2 通过 MPLAB ICE 4 运行单元测试

下面举例说明将 Unity 与硬件工具 MPLAB ICE 4 在线仿真器配合使用的操作步骤。

1.5.2.1 步骤 1. 设置通过软件模拟器进行测试和验证的项目

建议先通过软件模拟器进行测试，然后再继续使用 MPLAB ICE 4 在线仿真器。因此，首先按照 1.5.1. 通过软件模拟器运行单元测试所述步骤操作。确保您的项目具有测试运行程序和串行驱动程序并且使能了 printf 支持。

通过软件模拟器执行测试后，便可继续使用 MPLAB ICE 4。要通过仿真器捕捉串行输出，需使用下表中的通信设置。

串行驱动程序设置	所需值
波特率	115200
发送位	8
接收位	8
数据奇偶校验	无奇偶校验

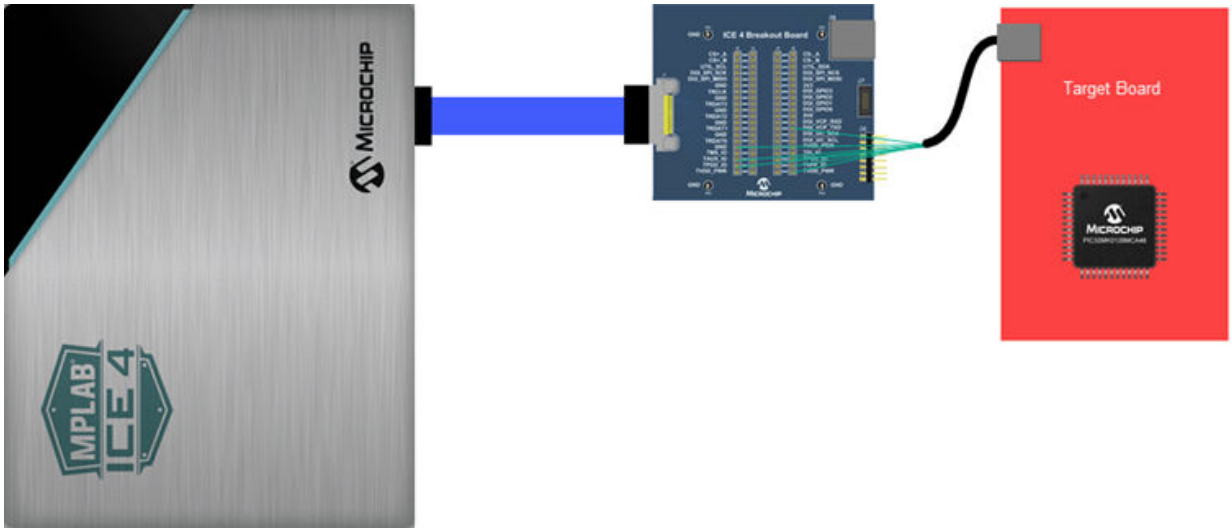
1.5.2.2 步骤 2. 使用转接板将 MPLAB ICE 4 连接到目标

使用 ICE 4 转接板（AC244141）将 MPLAB ICE 4 连接到硬件。调试和编程器件时需要 ICSP 引脚，读取串行输出时需要 DGI_VCP_TXD 引脚。DGI_VCP_TXD 需要连接到配置为供串行驱动程序使用的引脚。

表 1-1. 连接汇总

ICE 4 转接板引脚名称	目标
TVPP_IO	ICSP 连接器 1
TVDD_PWR	ICSP 连接器 2
GND	ICSP 连接器 3
TPGD_IO	ICSP 连接器 4
TPGC_IO	ICSP 连接器 5
TAUX_IO	ICSP 连接器 6
DGI_VCP_TXD	供配置的串行驱动程序使用的 TX 引脚

图 1-7. 硬件连接



1.5.2.3 步骤 3. 使用本地 ICE 4 验证连接

使用 USB 将 MPLAB ICE 4 连接到计算机。打开并使用 MPLAB X IDE 验证能否使用仿真器和 ICE 4 转接板在目标上编程和运行测试应用程序。

如果不能，需要检查转接板连接。

注：目前无法获取串行输出。

1.5.2.4 步骤 4. 将 MPLAB ICE 4 配置为网络工具

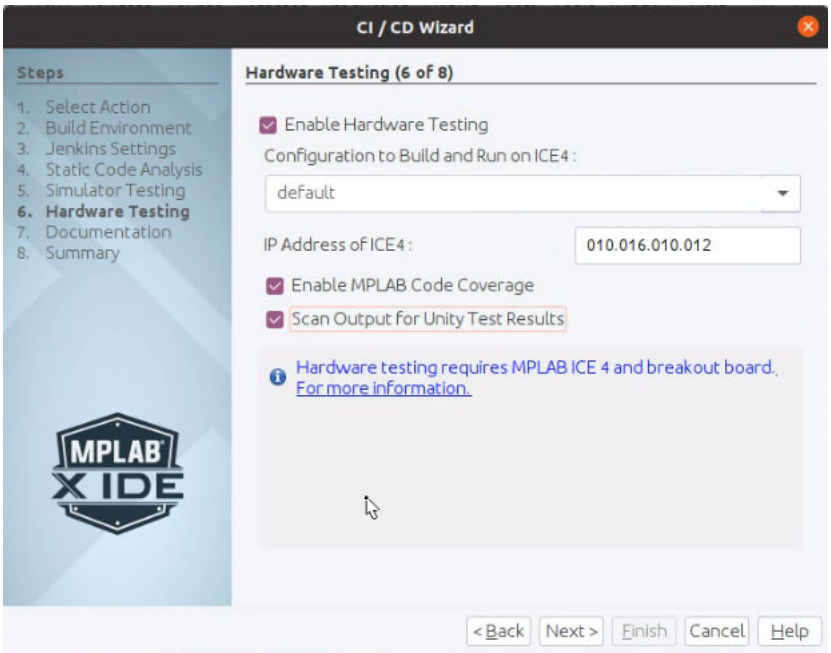
在 MPLAB X IDE 中，打开 *Tools > Manage Network Tools*，将连接的 MPLAB ICE 4 配置为使用以太网而非 USB。MPLAB ICE 4 需要连接到以太网线缆，因此需要断开 USB 线缆，之后切换 MPLAB ICE 4 电源。有关详细信息，请参见“*MPLAB ICE 4 在线仿真器用户指南*”。

完成配置后，扫描网络工具时，MPLAB ICE 4 应出现在 Manage Network Tools（管理网络工具）对话框中，此时应记下 IP 地址。要验证以太网设置，现在可以在 MPLAB X IDE 中使用 MPLAB ICE 4 作为远程工具来编程和运行测试应用程序。

注：如果网络未提供静态 IP 地址，则需要定期重复此步骤。

1.5.2.5 步骤 5. 在 CI/CD 流水线中添加 MPLAB ICE 4

在 CI/CD 向导中，现在应准备好使能 MPLAB ICE 4 硬件测试作为 Jenkins 编译流水线生成的一部分。



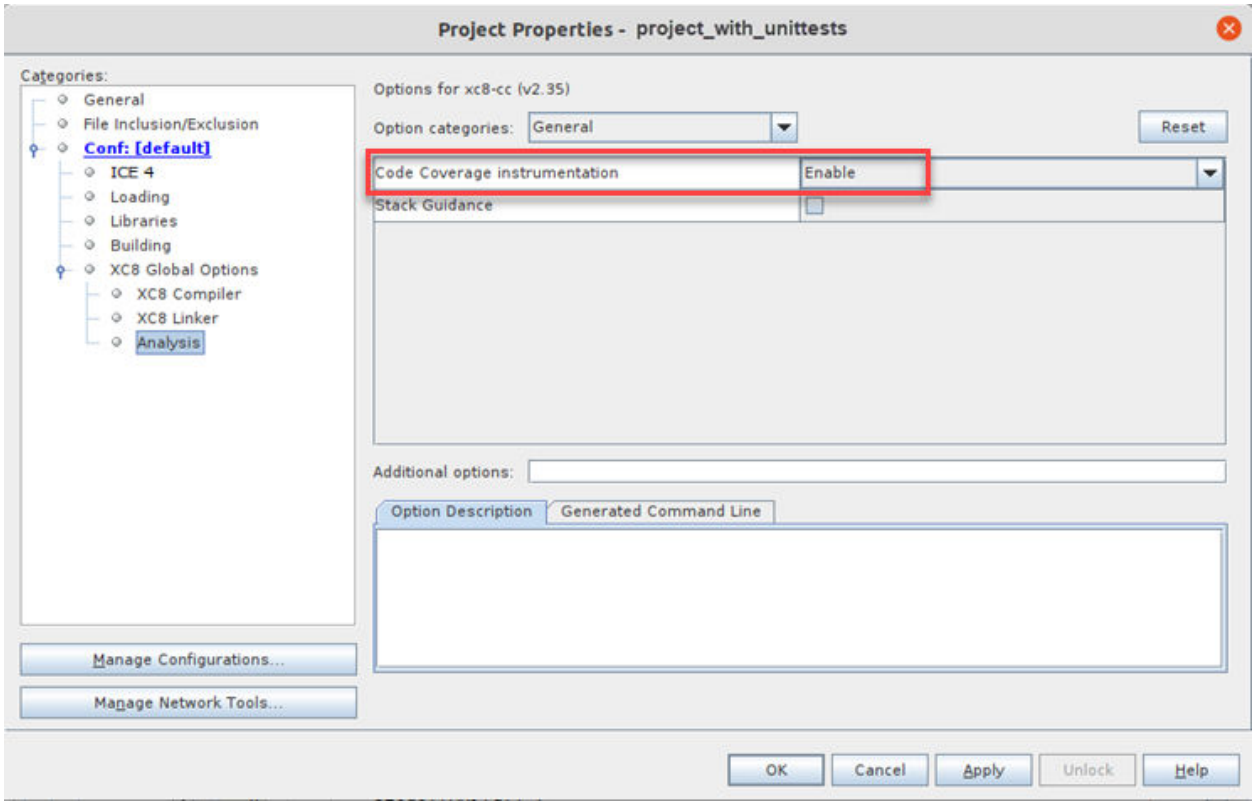
选择要测试的编译配置。

配置 MPLAB ICE 4 的以太网时，填入之前记下的 IP 地址。

如果需要覆盖数据，应使能 *MPLAB 代码覆盖*。但注意，这需要将项目设置为对编译进行代码覆盖插装。为此，需要打开项目属性，在 Analysis（分析）部分配置 Code Coverage instrumentation（代码覆盖插装）（见下图）。

使能 *Scan Output for Unity Test Results*，这将在 Jenkins 编译作业中解析串行端口的输出并生成测试报告。

图 1-8. 使能代码覆盖



完成所有配置后，生成的 Jenkins 编译作业将在运行时执行以下操作：

- 使用所选的编译配置编译项目
- 与 MPLAB ICE 4 通信并捕捉串行输出
- 使用生成的 MDB 脚本运行测试应用程序
- 完成后，解析串行输出并显示测试结果
- 发布覆盖数据

1.5.2.6 步骤 6. 更新 MPLAB ICE 4 IP 地址（可选）

如果后续迁移 MPLAB ICE 4 或者网络使用动态 IP 地址，可能需要使用新的 IP 地址更新编译脚本。

在 MPLAB X IDE 中，选择 *Tools > Manage Network Tools*，然后扫描工具。找到 MPLAB ICE 4，记下新的 IP 地址。

在 *Jenkinsfile* 中，更新 *HARDWARE_TEST_TOOL_IP* 字段：

```
HARDWARE_TEST_TOOL_IP =  '123.123.12.34'
```

在 *mdb-hardware-script.txt* 中，更新 *addip* 和 *hwtool* 命令：

```
# Add IP of tool so it can be detected
addip 123.123.012.034
# (Optional) List tools to verify that tool was detected
hwtool
# Connect to network configured tool
hwtool ICE4 <ipa>123.123.012.034
```

注：mdb 脚本使用零填充格式，因此使用 *123.123.012.034* 代替 *123.123.12.34*

1.6 Dockerfile 入门

Docker 是一个用于开发、交付和运行应用程序的开放式平台，可提供 OS 级虚拟化，能够将应用程序与基础架构分开。

Docker 将读取来自 Dockerfile 的指令，并编译 Docker 映像。Docker 映像类似模板，用于编译和运行 Docker 容器。一切的前提条件是必须已安装 Docker。

有关如何安装 Docker 和使用 Dockerfile 的更多信息，请参见 www.docker.com。

1.6.1 准备工作

1.6.1.1 许可证服务器

- 要将设置中的 MPLAB XC PRO 编译器与 Docker 容器配合使用，必须可从编译节点访问 MPLAB XC 网络许可证服务器。ww1.microchip.com/downloads/en/DeviceDoc/50002334B.pdf《MPLAB® XC 编译器许可证服务器手册》（DS50002334C_CN）介绍了如何设置该服务器。
- 其他功能可能需要由许可证服务器来验证其许可证，例如 MPLAB 分析工具套件（MISRA 检查和 MPLAB 代码覆盖）。

1.6.2 在 MPLAB X IDE 中生成 Dockerfile

1. 在 MPLAB X IDE 中打开项目，然后打开 *Tools > CI/CD Wizard*。
2. 选择 **Docker File**（Docker 文件）作为输出类型。
3. 根据需要在向导中填入剩余信息。请参见以下章节。
4. 生成 *Dockerfile* 后，验证其是否已添加到项目基本文件夹中。

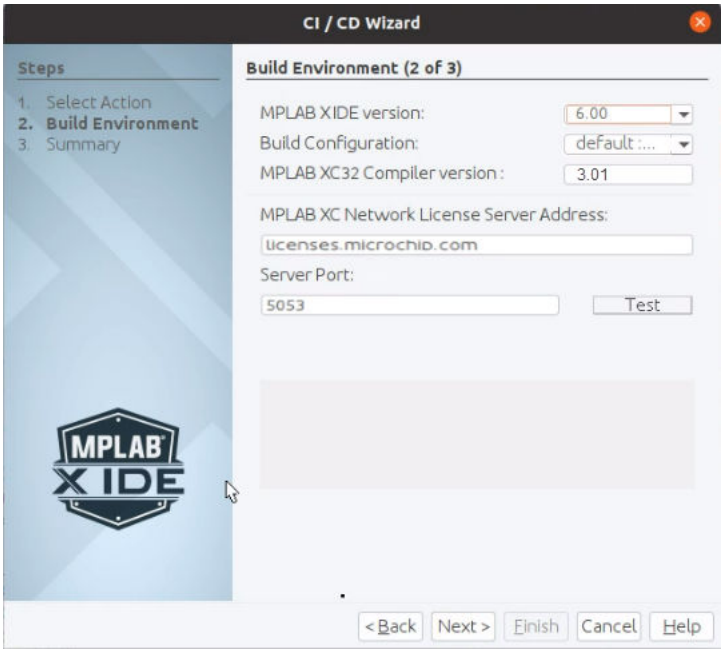
1.6.2.1 CI/CD 向导对话框 1——仅限 Docker

在本例中，选择“Docker File”。单击 **Next** 继续。

注：需拥有 MPLAB XC 编译器许可证才能创建 Docker 文件。

1.6.2.2 CI/CD 向导对话框 2——仅限 Docker

设置要在容器中使用的编译环境。如果拥有网络服务器许可证，则可以提供如何访问该许可证以进行编译的相关信息，否则将文本框留空（不过将出现一条说明）。



选项	说明
MPLAB IDE version	从下拉列表中选择支持的版本。默认为最新版本。
Build Configuration	选择编译所需的项目配置
MPLAB XC Compiler version	选择编译所需的编译器版本
MPLAB XC Network License Server Address	输入服务器的地址。例如：licenses.microchip.com

.....（续）	
选项	说明
Server Port	输入服务器的端口号。例如： 5053
信息性说明（视情况提供）	如果拥有工作站许可证，将显示有关购买网络服务器许可证的信息。

1.6.2.3 CI/CD 向导对话框 3——仅限 Docker

查看汇总可发现将仅生成 Dockerfile。

单击 **Back** 可更改设置项。单击 **Finish** 可完成设置。CI/CD 向导随后将根据设置信息生成 Dockerfile。

1.6.2.4 CI/CD 向导输出——仅限 Docker

在 MPLAB X IDE 中，可以看到以下内容：

1. Output 窗口中正在生成的文件。
2. 生成后添加到项目文件夹中的文件。
3. WebHelp（在单独窗口中）。
4. Dockerfile（在单独窗口中）。

该输出是 Jenkins 输出的子集。

1.6.3 生成的 Dockerfile 的内容

生成的 Dockerfile 是针对具体的 MPLAB X IDE 项目而定制的。

设置包括：

- Debian GNU/Linux 10
- *curl*、*make* 和 *unzip* 等基本工具
- MPLAB X IDE 安装（含二进制文件）：
  - *mdb*——Microchip 调试器
  - *prjMakefilesGenerator*——用于为 MPLAB X IDE 项目生成 Makefiles 的 CLI
  - *xclm*——XC 工具链许可证管理器
  - *IPE*——MPLAB 集成编程环境

- MPLAB XC 工具链安装。使用 MPLAB X IDE 项目中指定的版本
- MPLAB X IDE 项目中指定的器件系列包（Device Family Pack，DFP）

1.6.4 建议的后续步骤

生成的 Dockerfile 可用于为 MPLAB X IDE 项目设置编译环境。有关更多信息，请参见 Dockerfile 参考。

1.6.4.1 本地测试映像

dockerfile 便于本地创建和测试 docker 映像。如果尚未在本地安装 Docker，可从 www.docker.com 下载。

用于编译和运行映像的命令：

```
# Build an image based on the generated Dockerfile
docker build --tag mplabx-env --file Dockerfile .

# Check that the 'mplabx-env' image was created
docker images

# Start a docker container in interactive mode and map MPLAB X project source
docker run -it -v PATH_TO_MY_PROJECT_HERE:/usr/src/project mplabx-env bash
```

启动容器后，可以运行以下命令来编译映射的 MPLAB X IDE 项目：

```
# Navigate to the mapped MPLAB X project
cd /usr/src/project

# Generate Makefiles
prjMakefilesGenerator .

# Build project
make

# List files generated in the dist folder
find dist
```

1.6.4.2 在 CI/CD 框架中使用 Dockerfile

CI/CD 向导已内置 **Jenkins** 支持；可以为该框架生成支持文件。对于其他框架，需要参考相应文档来了解如何使用 Docker。

1.6.4.3 为 Dockerfile 扩展额外的工具/Linux 软件包

可通过修改 dockerfile 安装额外的 Linux 软件包。首先以交互模式启动映像，然后通过以下命令行手动安装软件包：

```
apt-get update
apt-get install [packageName]
```

确认软件包可用且正常工作后，可将其添加到 Dockerfile 中：

```
RUN apt-get update -yq \
 && apt-get install -yq --no-install-recommends \
 [packageName] \
 && apt-get clean && rm -rf /var/lib/apt/lists/* /tmp/* /var/tmp/*
```

注：最后一行清理代码用于尽可能减小 Docker 映像。

1.7 CICDW 命令行工具

如果希望运行 CI/CD 向导的命令行版本，可从`<MPLAB X IDE version install directory>mplab_platform/bin/cicdw.exe` 下获取。

使用 `cicdw.exe -h` 获取命令选项的帮助信息（见下文）。如需特定命令的帮助信息，应搭配使用该命令和`-h`；例如：`cicdw.exe generate-jenkins -h`。

注：所有命令选项必须位于同一行，如下所示：

```
cicdw.exe generate-jenkins --misra --simulator ...
```

cicdw 帮助

```
usage: cicdw [-h] [-v] [-q] [--version]
             {generate-jenkins,generate-docker,generate-doxyfile,container,jenkins-server}
             ...

CI/CD Wizard for MPLAB X projects

optional arguments:
  -h, --help            show this help message and exit
  -v                    Enable verbose logging (Use -vv for extra verbose
                        logging)
  -q                    Show less output.(Use -qq to only show errors and
                        warnings and -qqq to only show errors)
  --version             Print version information

Commands:
  {generate-jenkins,generate-docker,generate-doxyfile,container,jenkins-server}
    generate-jenkins    Generate jenkins build files (Jenkinsfile and
                        Dockerfile)
    generate-docker     Generate dockerfile
    generate-doxyfile   Generate Doxyfile for doxygen documentation
    container           Manage local build container
    jenkins-server      Manage jenkins server

Type {command} -h for command specific help
```

1.7.1 CICDW Jenkins 示例

粗体文本表示开发人员指定的内容。以下所有内容必须在同一行。

```
MPLABPath\mplab-nbplatform\nbbuild\netbeans\bin\cicdw.exe> generate-jenkins
--xclm ToolchainPath \xc8\v2.32\bin\xclm.exe
--configuration default
--mplabx-version 6.00
--toolchain-version 1.61
--license-server-name licenses.microchip.com
--license-server-port 5053
--docker-host-agent-label docker
--misra --use-cppcheck-trend
--simulate --simulate-with-coverage --simulate-with-unity --simulate-sercom-output uart1
 --simulate-build-configuration default
--doxygen --doxygen-configuration-file Doxyfile --doxygen-generate-doxyfile
--doxygen-project-name test --doxygen-input-encoding ISO-8859-1
--doc-artifact-html doc-html.zip --doc-artifact-latex doc-latex.zip --doc-artifact-pdf doc-
pdf.zip
--hardware-test --hardware-test-build-configuration default --hardware-test-tool-ip
192.168.16.3
--hardware-test-with-coverage --hardware-test-with-unity
ProjectPath
```

1.7.2 CICDW Docker 示例

粗体文本表示开发人员指定的内容。以下所有内容必须在同一行。

```
MPLABPath\mplab-nbplatform\nbbuild\netbeans\bin\cicdw.exe> generate-docker
--xclm ToolchainPath\xc8\v2.32\bin\xclm.exe
--configuration default
--mplabx-version 6.00
--toolchain-version 2.32
--license-server-name licenses.microchip.com
--license-server-port 5053
ProjectPath
```

Microchip 网站

Microchip 网站（www.microchip.com）为客户提供在线支持。客户可通过该网站方便地获取文件和信息。我们的网站提供以下内容：

- **产品支持**——数据手册和勘误表、应用笔记和示例程序、设计资源、用户指南以及硬件支持文档、最新的软件版本以及归档软件
- **一般技术支持**——常见问题解答（FAQ）、技术支持请求、在线讨论组以及 Microchip 设计伙伴计划成员名单
- **Microchip 业务**——产品选型和订购指南、最新 Microchip 新闻稿、研讨会和活动安排表、Microchip 销售办事处、代理商以及工厂代表列表

产品变更通知服务

Microchip 的产品变更通知服务有助于客户了解 Microchip 产品的最新信息。注册客户可在他们感兴趣的某个产品系列或开发工具发生变更、更新、发布新版本或勘误表时，收到电子邮件通知。

欲注册，请访问 www.microchip.com/pcn，然后按照注册说明进行操作。

客户支持

Microchip 产品的用户可通过以下渠道获得帮助：

- 代理商或代表
- 当地销售办事处
- 应用工程师（ESE）
- 技术支持

客户应联系其代理商、代表或 ESE 寻求支持。当地销售办事处也可为客户提供帮助。本文档后附有销售办事处的联系方式。

也可通过 www.microchip.com/support 获得网上技术支持。

Microchip 器件代码保护功能

请注意以下有关 Microchip 产品代码保护功能的要点：

- Microchip 的产品均达到 Microchip 数据手册中所述的技术规范。
- Microchip 确信：在正常使用且符合工作规范的情况下，Microchip 系列产品非常安全。
- Microchip 注重并积极保护其知识产权。严禁任何试图破坏 Microchip 产品代码保护功能的行为，这种行为可能会违反《数字千年版权法案》（Digital Millennium Copyright Act）。
- Microchip 或任何其他半导体厂商均无法保证其代码的安全性。代码保护并不意味着我们保证产品是“牢不可破”的。代码保护功能处于持续发展中。Microchip 承诺将不断改进产品的代码保护功能。

法律声明

提供本文档的中文版本仅为了便于理解。请勿忽视文档中包含的英文部分，因为其中提供了有关 Microchip 产品性能和使用情况的有用信息。Microchip Technology Inc.及其分公司和相关公司、各级主管与员工及事务代理机构对译文中可能存在的任何差错不承担任何责任。建议参考 Microchip Technology Inc.的英文原版文档。

本出版物及其提供的信息仅适用于 Microchip 产品，包括设计、测试以及将 Microchip 产品集成到您的应用中。以其他任何方式使用这些信息都将被视为违反条款。本出版物中的器件应用信息仅为您提供便利，将来可能会发生更新。如需额外的支持，请联系当地的 Microchip 销售办事处，或访问 https://www.microchip.com/en-us/support/design-help/client-supportservices。

Microchip“按原样”提供这些信息。Microchip 对这些信息不作任何明示或暗示、书面或口头、法定或其他形式的声明或担保，包括但不限于针对非侵权性、适销性和特定用途的适用性的暗示担保，或针对其使用情况、质量或性能的担保。

在任何情况下，对于因这些信息或使用这些信息而产生的任何间接的、特殊的、惩罚性的、偶然的或间接的损失、损害或任何类型的开销，Microchip 概不承担任何责任，即使 Microchip 已被告知可能发生损害或损害可以预见。在法律允许的最大范围内，对于因这些信息或使用这些信息而产生的所有索赔，Microchip 在任何情况下所承担的全部责任均不超出您为获得这些信息向 Microchip 直接支付的金额（如有）。如果将 Microchip 器件用于生命维持和/或生命安全应用，一切风险由买方自负。买方同意在由此引发任何一切损害、索赔、诉讼或费用时，会维护和保障 Microchip 免于承担法律责任。除非另外声明，在 Microchip 知识产权保护下，不得暗中或以其他方式转让任何许可证。

商标

Microchip 的名称和徽标组合、Microchip 徽标、Adaptec、AVR、AVR 徽标、AVR Freaks、BesTime、BitCloud、CryptoMemory、CryptoRF、dsPIC、flexPWR、HELDO、IGLOO、JukeBlox、KeeLoq、Kleer、LANCheck、LinkMD、maXStylus、maXTouch、MediaLB、megaAVR、Microsemi、Microsemi 徽标、MOST、MOST 徽标、MPLAB、OptoLyzer、PIC、picoPower、PICSTART、PIC32 徽标、PolarFire、Prochip Designer、QTouch、SAM-BA、SenGenuity、SpyNIC、SST、SST 徽标、SuperFlash、Symmetricom、SyncServer、Tachyon、TimeSource、tinyAVR、UNI/O、Vectron 及 XMEGA 均为 Microchip Technology Incorporated 在美国和其他国家或地区的注册商标。

AgileSwitch、APT、ClockWorks、The Embedded Control Solutions Company、EtherSynch、Flashtec、Hyper Speed Control、HyperLight Load、Libero、motorBench、mTouch、Powermite 3、Precision Edge、ProASIC、ProASIC Plus、ProASIC Plus 徽标、Quiet-Wire、SmartFusion、SyncWorld、Temux、TimeCesium、TimeHub、TimePictra、TimeProvider、TrueTime 和 ZL 均为 Microchip Technology Incorporated 在美国的注册商标。

Adjacent Key Suppression、AKS、Analog-for-the-Digital Age、Any Capacitor、AnyIn、AnyOut、Augmented Switching、BlueSky、BodyCom、Clockstudio、CodeGuard、CryptoAuthentication、CryptoAutomotive、CryptoCompanion、CryptoController、dsPICDEM、dsPICDEM.net、Dynamic Average Matching、DAM、ECAN、Espresso T1S、EtherGREEN、GridTime、IdealBridge、In-Circuit Serial Programming、ICSP、INICnet、Intelligent Paralleling、IntelliMOS、Inter-Chip Connectivity、JitterBlocker、Knob-on-Display、KoD、maxCrypto、maxView、memBrain、Mindi、MiWi、MPASM、MPF、MPLAB Certified 徽标、MPLIB、MPLINK、MultiTRAK、NetDetach、Omniscient Code Generation、PICDEM、PICDEM.net、PICkit、PICtail、PowerSmart、PureSilicon、QMatrix、REAL ICE、Ripple Blocker、RTAX、RTG4、SAM-ICE、Serial Quad I/O、simpleMAP、SimpliPHY、SmartBuffer、SmartHLS、SMART-I.S.、storClad、SQI、SuperSwitcher、SuperSwitcher II、Switchtec、SynchroPHY、Total Endurance、Trusted Time、TSHARC、USBCheck、VariSense、VectorBlox、VeriPHY、ViewSpan、WiperLock、XpressConnect 和 ZENA 均为 Microchip Technology Incorporated 在美国和其他国家或地区的商标。

SQTP 为 Microchip Technology Incorporated 在美国的服务标记。

Adaptec 徽标、Frequency on Demand、Silicon Storage Technology 和 Symmcom 均为 Microchip Technology Inc.在除美国外的国家或地区的注册商标。

GestIC 为 Microchip Technology Inc.的子公司 Microchip Technology Germany II GmbH & Co. KG 在除美国外的国家或地区的注册商标。

在此提及的所有其他商标均为各持有公司所有。

© 2023, Microchip Technology Incorporated 及其子公司版权所有。

ISBN： 978-1-6683-1383-1

质量管理体系

有关 Microchip 的质量管理体系的信息，请访问 www.microchip.com/quality。

全球销售及服务网点

美洲	亚太地区	亚太地区	欧洲
公司总部 2355 West Chandler Blvd. Chandler, AZ 85224-6199 电话：480-792-7200 传真：480-792-7277 技术支持： www.microchip.com/support 网址： www.microchip.com **亚特兰大** 德卢斯，佐治亚州 电话：678-957-9614 传真：678-957-1455 **奥斯汀，德克萨斯州** 电话：512-257-3370 **波士顿** 韦斯特伯鲁，马萨诸塞州 电话：774-760-0087 传真：774-760-0088 **芝加哥** 艾塔斯卡，伊利诺伊州 电话：630-285-0071 传真：630-285-0075 **达拉斯** 阿迪森，德克萨斯州 电话：972-818-7423 传真：972-818-2924 **底特律** 诺维，密歇根州 电话：248-848-4000 **休斯顿，德克萨斯州** 电话：281-894-5983 **印第安纳波利斯** 诺布尔斯维尔，印第安纳州 电话：317-773-8323 传真：317-773-5453 电话：317-536-2380 **洛杉矶** 米慎维荷，加利福尼亚州 电话：949-462-9523 传真：949-462-9608 电话：951-273-7800 **罗利，北卡罗来纳州** 电话：919-844-7510 **纽约，纽约州** 电话：631-435-6000 **圣何塞，加利福尼亚州** 电话：408-735-9110 电话：408-436-4270 **加拿大 - 多伦多** 电话：905-695-1980 传真：905-695-2078	**澳大利亚 - 悉尼** 电话：61-2-9868-6733 **中国 - 北京** 电话：86-10-8569-7000 **中国 - 成都** 电话：86-28-8665-5511 **中国 - 重庆** 电话：86-23-8980-9588 **中国 - 东莞** 电话：86-769-8702-9880 **中国 - 广州** 电话：86-20-8755-8029 **中国 - 杭州** 电话：86-571-8792-8115 **中国 - 香港特别行政区** 电话：852-2943-5100 **中国 - 南京** 电话：86-25-8473-2460 **中国 - 青岛** 电话：86-532-8502-7355 **中国 - 上海** 电话：86-21-3326-8000 **中国 - 沈阳** 电话：86-24-2334-2829 **中国 - 深圳** 电话：86-755-8864-2200 **中国 - 苏州** 电话：86-186-6233-1526 **中国 - 武汉** 电话：86-27-5980-5300 **中国 - 西安** 电话：86-29-8833-7252 **中国 - 厦门** 电话：86-592-2388138 **中国 - 珠海** 电话：86-756-3210040	**印度 - 班加罗尔** 电话：91-80-3090-4444 **印度 - 新德里** 电话：91-11-4160-8631 **印度 - 浦那** 电话：91-20-4121-0141 **日本 - 大阪** 电话：81-6-6152-7160 **日本 - 东京** 电话：81-3-6880-3770 **韩国 - 大邱** 电话：82-53-744-4301 **韩国 - 首尔** 电话：82-2-554-7200 **马来西亚 - 吉隆坡** 电话：60-3-7651-7906 **马来西亚 - 槟榔屿** 电话：60-4-227-8870 **菲律宾 - 马尼拉** 电话：63-2-634-9065 **新加坡** 电话：65-6334-8870 **台湾地区 - 新竹** 电话：886-3-577-8366 **台湾地区 - 高雄** 电话：886-7-213-7830 **台湾地区 - 台北** 电话：886-2-2508-8600 **泰国 - 曼谷** 电话：66-2-694-1351 **越南 - 胡志明市** 电话：84-28-5448-2100	**奥地利 - 韦尔斯** 电话：43-7242-2244-39 传真：43-7242-2244-393 **丹麦 - 哥本哈根** 电话：45-4485-5910 传真：45-4485-2829 **芬兰 - 埃斯波** 电话：358-9-4520-820 **法国 - 巴黎** 电话：33-1-69-53-63-20 传真：33-1-69-30-90-79 **德国 - 加兴** 电话：49-8931-9700 **德国 - 哈恩** 电话：49-2129-3766400 **德国 - 海尔布隆** 电话：49-7131-72400 **德国 - 卡尔斯鲁厄** 电话：49-721-625370 **德国 - 慕尼黑** 电话：49-89-627-144-0 传真：49-89-627-144-44 **德国 - 罗森海姆** 电话：49-8031-354-560 **以色列 - 若那那市** 电话：972-9-744-7705 **意大利 - 米兰** 电话：39-0331-742611 传真：39-0331-466781 **意大利 - 帕多瓦** 电话：39-049-7625286 **荷兰 - 德卢内市** 电话：31-416-690399 传真：31-416-690340 **挪威 - 特隆赫姆** 电话：47-72884388 **波兰 - 华沙** 电话：48-22-3325737 **罗马尼亚 - 布加勒斯特** 电话：40-21-407-87-50 **西班牙 - 马德里** 电话：34-91-708-08-90 传真：34-91-708-08-91 **瑞典 - 哥德堡** 电话：46-31-704-60-40 **瑞典 - 斯德哥尔摩** 电话：46-8-5090-4654 **英国 - 沃金厄姆** 电话：44-118-921-5800 传真：44-118-921-5820