在 MCUXpresso IDE v11.0.0 中使用 LPC55S69 SDK Trustzone 示例

由 LPCX presso support 员工 创建于 2019-6-12 的博客文章

原文：

https://community.nxp.com/community/mcuxpresso/mcuxpresso-ide/blog/2019/06/12/using-lpc55s69-sdk-trustzone-examples-with-mcuxpresso-ide-v1100

这篇文章描述了搭载 MCUXpresso IDE 的 LPC55S69 MCU 所使用的 SDK Trustzone 的例子（Hello World）。

阅读这篇博客需要了解以下内容:

1. MCUXpresso IDE 11.0x 版
2. LPCXpresso 55S69 EVK 开发板
3. LPCXpresso 55S69 EVK 2.6 版 SDK

旧版的 MCUXpresso IDE 和 SDK 中的 trustzone 例程性能较差，因此不推荐使用。

对 ARM Trustzone 概念的理解也将有助于理解本文中所示的特性。请另参阅 NXP 社区中有关 LPC55S6x 的 Trustzone 的文章。

最后，这里假设您已熟悉了 MCUXpresso IDE 的使用和 SDK 的安装。

概述

本质上，结合 ARM Trustzone 技术的 MCU 能够在安全区和非安全区之间提供系统级的隔离。在非安全的环境中，对某些操作、外围设备等的访问由安全区控制。安全区和非安全区的互访，这些都可由单个 CPU 在可执行时间内实现。

使用 Trustzone 功能开发系统带来了许多挑战，特别是为非安全应用程序开发人员提供 Secure 开发服务，调试安全和非安全代码（工程）以及在 IDE 中提供易于使用和理解的使用模型。

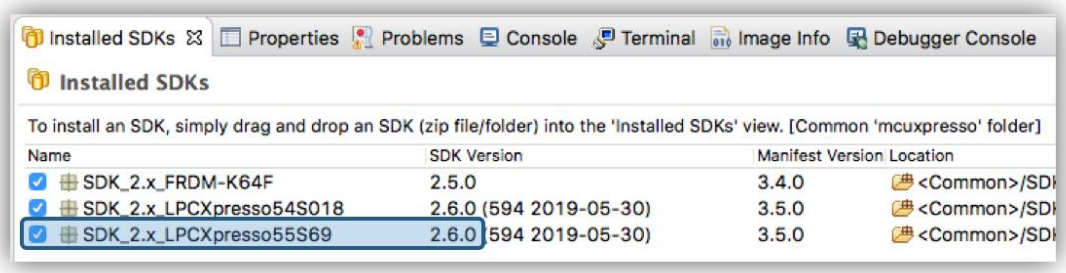
SDK 附带的例程包含安全和非安全可链接工程，并在 IDE 中进行配置以在安全和非安全开发的角度演示开发。

但是，更典型的“现实世界”情况可能是开发人员从事非安全工程开发并将安全区视为提供大量非常安全操作的“黑匣子”（或引导加载程序）。 我们将在后续文章中讨论这些使用场景。

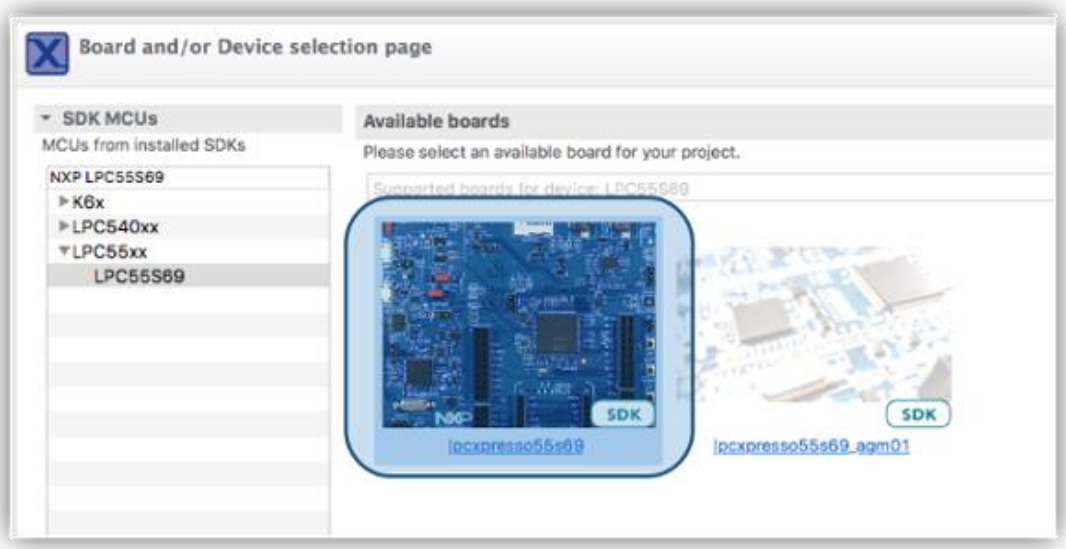
注意：LPC55S69 是一个多核 MCU，因为它包含两个独立的 CPU - 编号为 0 和 1.在这些示例中，所有代码都针对 CPU 0，CPU 1 的使用超出了本文的论述范围。

导入（链接）例程

首先，确保已经将 LPCXpresso55S69 EVK 板的 SDK 版本 2.6 安装到 MCUXpresso IDE 上。 安装完成后，将在 Installed SDKs 视图中显示它，其版本如下所示：

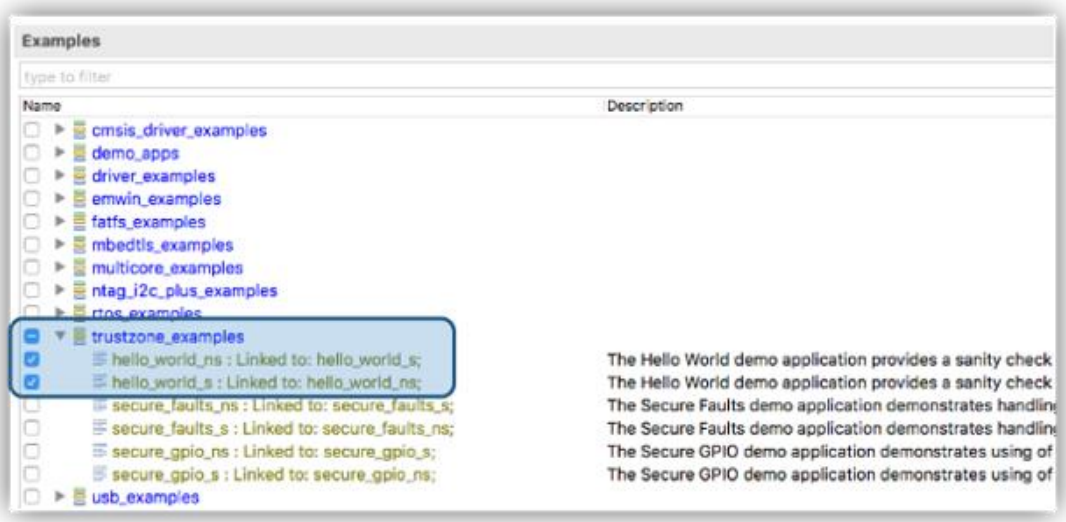


在 Quickstart 视图中，启动'Import SDK example（s）'向导，然后选择 LPCXpresso55S69 板并单击<next>



展开 trustzone 例程，然后单击任意一个 Hello World 项目。 由于这些工程是可链接工程，安全（_s）和非安全（_ns）例程都将被选中。 单击<finish>开始导入。

注意：可链接的工程在向导中显示为黄色。 设置此工程链接是因为非安全工程依赖于安全工程来访问允许的安全操作（调用）。安全工程引用非安全工程进行调试操作（flash 编程）。



导入后，您将看到 Project Explorer 视图已包含这些工程。

工程设置的说明

内存配置

通过打开工程设置然后打开内存设置，可以在 Project Explorer 视图中看到每个工程的内存分配。 从这里可以看到管理链接器脚本机制设置的输入内存范围（对于每个工程）：

除了 venee table 之外，这些工程可以被视为单独的主体，并且必须具有不重叠的存储区域。

Secure：地址空间安全的一个特征是，地址位【28】的设置表示安全位置（最高有效半字节中的奇数）。

0x10000000 处的闪存区域 PROGRAM_FLASH 指定安全的 LinkServer 闪存驱动程序 LPC55xx_S.cfx。

0x1000FE00 处的 SG_veneer_table 块用于代码，为（某些）安全功能提供网关并继承相同的闪存驱动程序。 注意：SG_veneer_table 是一个被 IDE 识别为 veneer 放置选项的“魔术”字符串（见下文）。

Non Secure：对于这些地址，未设置位【28】表示非安全位置（最高有效半字节中的偶数）

0x10000 处的闪存区域 PROGRAM_FLASH 指定非安全 LinkServer 闪存驱动程序 LPC55xx.cfx。

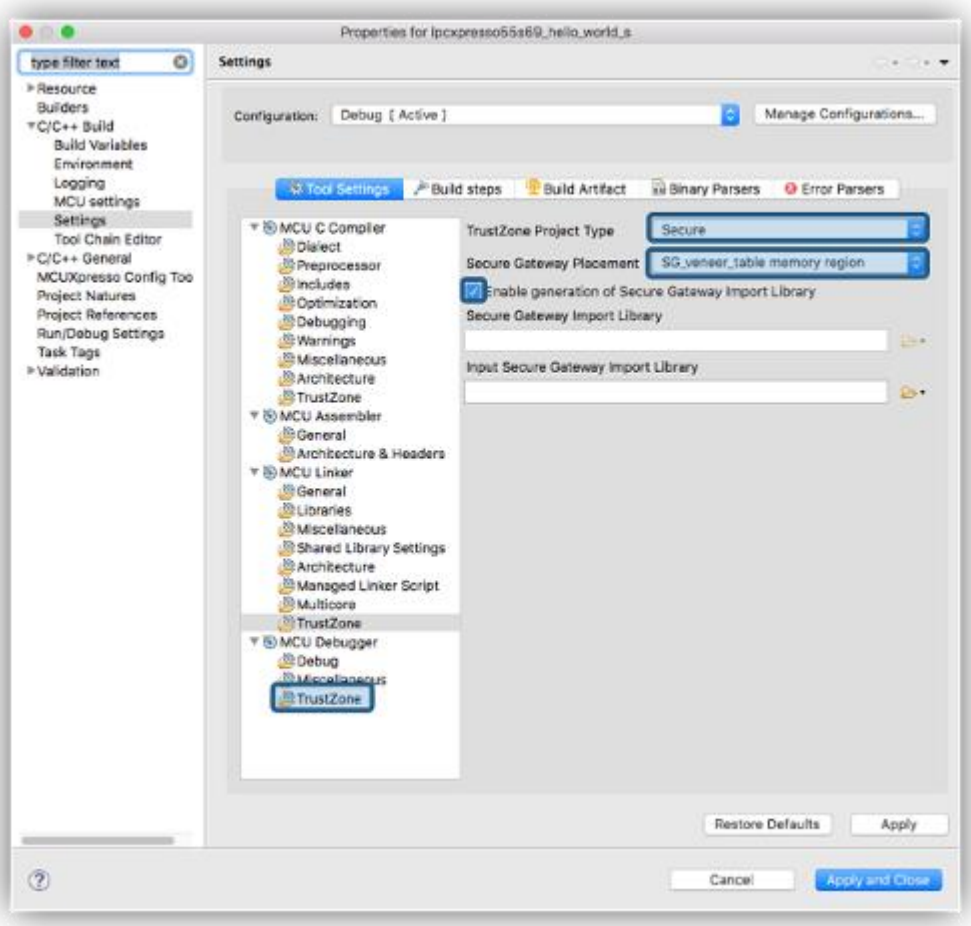
工程属性

支持 Trustzone 的 MCU 具有许多专用的**工程属性**，主要用于处理链接和调试。

设置为“安全”的工程将提供更多选项，包括安全 veneer table 位置的设置，以及是否生成导入库的复选框。 还可以指定其他 Secure Gateway（安全网关）导入库。

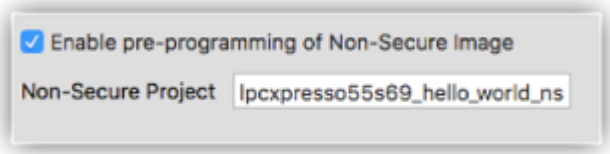
Secure（安全）：下面显示了 **MCU Linker - > Trustzone** 属性对话框。

请注意，安全网关放置（共享安全操作的安全 veneers 代码）设置为使用“魔术”闪存区域（在上一节中提到）进行放置。

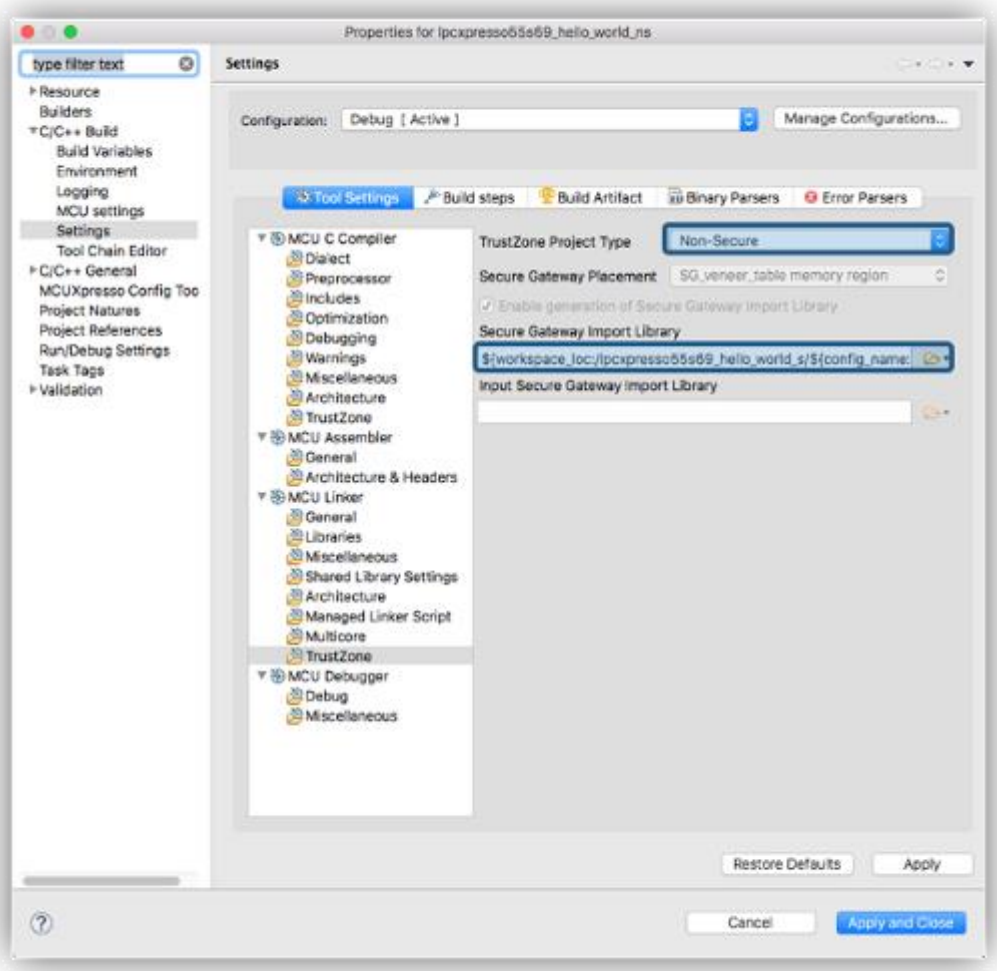


另请参阅 **MCU Debugger - > Trustzone** 选项。 如果使用（如本例所示），则获取相关的非安全项目的名称，然后在执行安全项目调试操作时自动（构建和）编程。 此外，还将添加此关联项目的符号，以允许两个项目的源级调试。

选中后，它将如下所示：



Non Secure（非安全）：下面显示了 **MCU Linker - > Trustzone** 属性对话框：注意 Secure Gateway Import Library（安全网关导入库）引用了 Secure projects（安全工程）生成的库位置。



编译示例项目

编译示例就像选择项目并单击 Quickstart 面板中的 Build 一样简单。 请注意，编译非安全工程将强制首先构建 Secure 项目以提供与 Secure Gateway 库的链接。

假设之前没有编译任何工程，编译非安全 Hello World 项目将导致生成安全和非安全.af（s）并生成如下输出：

内存区域	使用大小	区域大小	%占用百分比
Program Flash:	16872 B	65024 B	25.95%
SG_Venee_Table:	32 B	512 B	6.25%
RAM0:	4388 B	32 KB	13.39%

完成编译目标文件：`lpcxpresso55s69_hello_world_s.axf`

内存区域	使用大小	区域大小	%占用百分比
Program Flash:	7372 B	456 KB	1.58%
RAM0:	8448 B	172 KB	4.80%

完成编译目标文件：lpcxpresso55s69_hello_world_ns.axf

注意:编译安全工程不会强制编译非安全工程，因为在此方向上没有编译依赖项。

调试示例工程

如上文所述，这些示例从安全和非安全工程的角度演示了如何开发。 因此，初始流程旨在显示安全状态内的启动，然后迁移到将执行安全操作的非安全状态。

因此，首先在 Project Explorer 视图中选择 Secure Project（安全工程），然后单击 Quickstart 面板中的 Debug。 由于我们将选项设置为启用非安全二进制文件的预编程，因此调试操作将分为两个阶段。

注意：如果这是第一次调试此项目，您将看到选择调试工具的选项。 然后，由于这是一个多核 MCU，您还必须选择要使用的内核（确保将其设置为 Core 0）。

首先，非安全的二进制文件将被烧录到闪存中——这个自动步骤将像正常的二进制烧录操作一样进行，并生成一个与非安全工程相关的日志，如下所示:

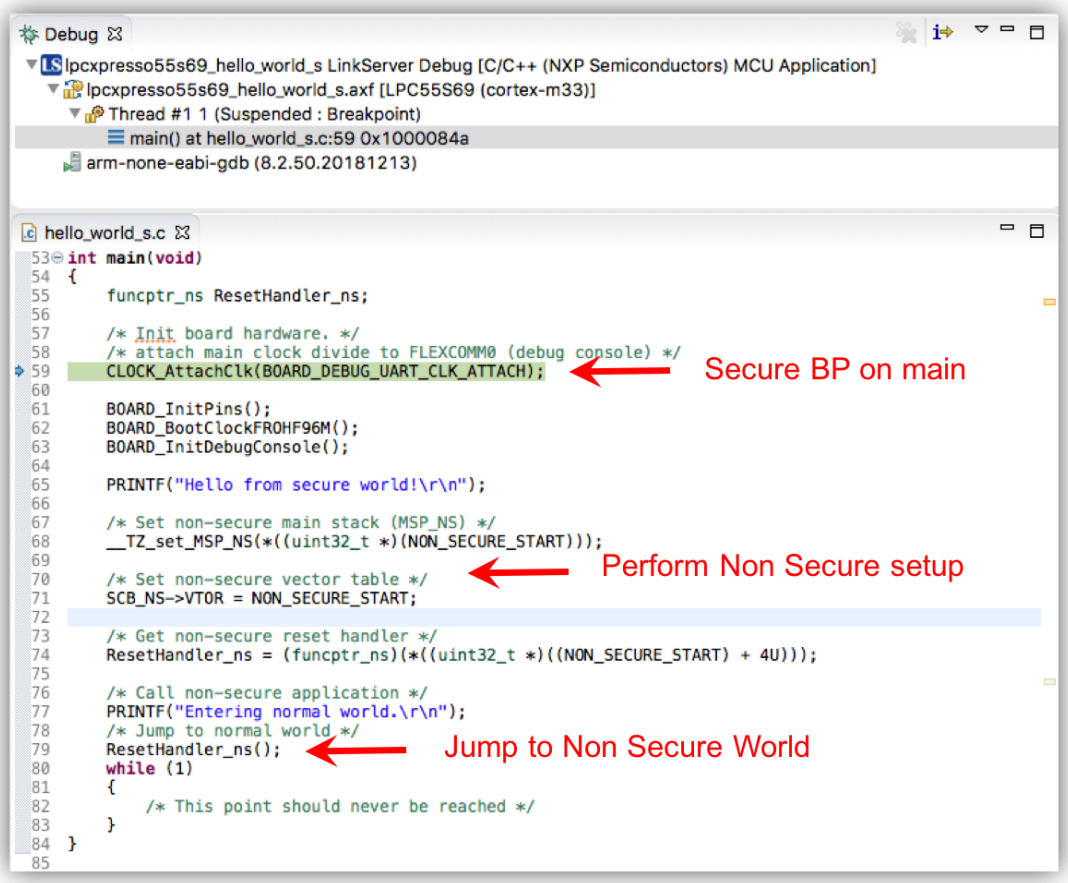
```
Flash variant 'LPC55xx (608KB)' detected (608KB = 19*32K at 0x0)
Writing 7372 bytes to address 0x00010000 in Flash
1 of 1 (   0) Writing pages 2-2 at 0x00010000 with 7372 bytes
(   0) at 00010000: 0 bytes - 0/7372
(100) at 00010000: 8192 bytes - 8192/7372
Erased/Wrote sector   2-2 with 7372 bytes in 122msec
Closing flash driver LPC55xx.cfx
(100) Finished writing Flash successfully.
Flash Write Done
Loaded 0x1CCC bytes in 417ms (about 17kB/s)
```

其次，安全工程将以正常方式烧录，这将生成与安全工程相关的标准调试日志，片段如下所示：

注意：如果这是第一次操作，则必须选择调试工具和 MCU 内核。 对于后续调试操作，这些选项将被记住。

Flash variant 'LPC55xx (608KB) (Secure)' detected (608KB = 19*32K at 0x10000000)
Writing 32 bytes to address 0x1000FE00 in Flash
1000A000 done 25% (8192 out of 32288)
1000C000 done 50% (16384 out of 32288)
1000E000 done 76% (24576 out of 32288)
10010000 done 100% (32768 out of 32288)
Erased/Wrote sector 1-1 with 32 bytes in 709msec
Closing flash driver LPC55xx_S.cfx
Flash Write Done
Flash Program Summary: 16904 bytes in 1.00 seconds (16.59 KB/sec)
Starting execution using system reset and halt target with a stall address
Retask read watchpoint 1 at 0x50000040 for boot ROM stall
Restore internally retasked watch #1 no type address 0x0 (unused)
Note - system reset leaves VTOR at 0x13000000 (not 0x10000000 which a booted image might assume)
Stopped (Was Reset) [Reset from Unknown]
Stopped: Breakpoint #2

烧录完成后，您应该在安全工程中的 main 上找到默认的 Breakpoint，如下所示：



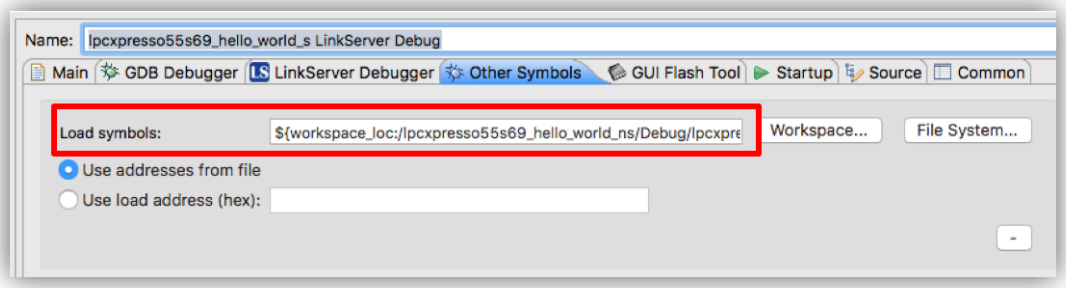
单击“运行”，此工程将首先将执行一些板/ MCU 初始化，打印问候语（如下所示），然后设置环境以执行非安全工程。

注意：这一步是必要的，因为非安全工程不是从复位运行的，因此不能依赖默认的硬件机制来初始化堆栈，中断处理程序等。这基本上与 bootloader 执行的操作相同。

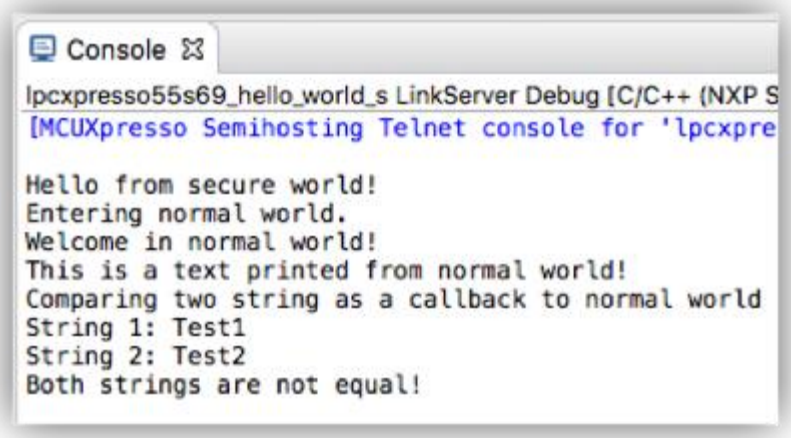
接下来，将跳转到非安全状态。

注意：由于非安全工程未直接调试，因此其主函数上没有默认断点。 要实现这个操作，建议打开非安全工程 Source - > hello_world_ns.c 并手动设置断点，如下所示：

提醒一下，因为我们有选项设置为启用非安全二进制文件的预编程，这也启用了非安全符号的加载（可以在安全工程调试启动配置中观察到（如下所示））。当我们实际调试安全工程时，正是加载这些符号使非安全工程的源代码级调试成为可能。



最后，单击 Run 运行此示例到终端窗口来观察输出结果。



本例程所演示的是非安全工程通过安全入口（NSE）访问安全区域，并调用安全工程的字符串比较操作函数。

这是此例程的最后一步。

另一种操作模式是直接在非安全工程上调试。 如果安全工程首先被烧录到闪存中，这是一个完全有效的操作。 如果是这样，非安全工程将像标准工程一样运行并停留在 main（）上的断点… **但是，如果没有预加载安全工程，调试操作将失败。**

如果调试操作失败…

可以将镜像文件烧录到闪存中，运行时可防止进一步的调试操作失败。如果发生这种情况，请按照以下步骤操作：

- 单击主 IDE 工具栏上的“清除调试”按钮——这将终止任何陈旧的低级调试操作
- 断开 USB 电缆与电路板的连接以断开电源
- 在 ISP 接头(J10)上放置一个跳线帽——这将强制 MCU 进入 ISP 模式而不是在 flash 中启动应用代码
- 现在将 USB 电缆重新连接到调试接口（P6）

- 现在，在 Project Explorer 视窗中选择一个工程，然后单击主 IDE 工具栏上的 GUI Flash Tool - 并使用它来整体擦除闪存。 [有关使用 GUI Flash Tools 的更多详细信息，请参阅随附的 IDE v11.0 用户指南]

如果成功，那么:
- 断开 USB 电缆与电路板的连接
- 移除 ISP 接头上的跳线帽(J10)
- 现在重新连接 USB 电缆到调试接口(P6)

现在尝试使用 Quickstart 面板中的 Debug 按钮再次为工程启动调试会话。 如果在应用程序开发过程中出现此问题，请尝试烧录一个已知的成功例程，以确保没有其他潜在的问题。