



# XMC MCU

## 32 位 微处理器

# XMC1000/4000 系列

**ARM™** ARM and Cortex-M are the trademarks of ARM Limited in the EU and other countries.

## XMC MCU 入门和应用指南

此文的内容是英飞凌 XMC 工程师多年工作的总结，但也仅代表工程师个人的观点，其中难免会有些错误，希望读者在发现错误后批评指正，反馈地址如下：

[chi.zhang@infineon.com](mailto:chi.zhang@infineon.com)

关于XMC 微处理器的相关资料，可访问如下网址：

<http://www.infineon.com/xmc>

## 修改记录

日期	版本	修改记录
2016-07-12	V1.0	

## 目录

修改记录.....	2
<b>1. 快速入门.....</b>	<b>5</b>
<b>2. 项目开发流程.....</b>	<b>11</b>
2.1 流程概述.....	11
2.2 项目开发实例.....	13
<b>3. 产品介绍.....</b>	<b>26</b>
3.1 产品路线.....	26
3.2 XMC1000 系列.....	27
3.2.1 产品特性.....	27
3.2.2 产品阵营.....	28
3.2.3 产品应用领域.....	28
3.3 XMC4000 系列.....	29
3.3.1 产品特性.....	29
3.3.2 产品阵营.....	30
3.3.3 产品应用领域.....	30
3.4 基于 XMC 的方案优势.....	31
3.4.1 电机控制方案.....	31
3.4.2 LED 控制方案.....	35
3.4.3 EtherCAT 控制方案.....	38
<b>4 集成开发环境 (IDE).....</b>	<b>40</b>
4.1 DAVE4.....	40
4.4.1 安装.....	40
4.4.2 新建工程.....	46
4.4.3 导入工程.....	47
4.4.4 调试技巧.....	49
4.4.5 使用 APP 流程.....	54
4.4.6 读取和修改 BMI.....	54
4.4.7 快捷键.....	55
4.4.8 文件比较功能.....	55
4.4.9 文件拷贝功能.....	56
4.4.10 代码和变量定位.....	56

4.4.11	查看程序运行周期数 .....	60
4.4.12	DAVE 工程转 Keil 工程流程 .....	60
4.2	Keil MDK .....	65
4.2.1	下载 .....	65
4.2.2	新建工程 .....	65
4.2.3	打开自带例程 .....	68
<b>5</b>	<b>调试工具 .....</b>	<b>70</b>
5.1	英飞凌板载 J-Link .....	70
5.1.1	XMC1000 板载 J-Link .....	70
5.1.2	XMC4000 板载 J-Link .....	71
5.2	Segger J-link .....	72
5.3	XMC-Link .....	72
5.4	J-Link 命令行 .....	73
5.4.1	擦除 Flash .....	74
5.4.2	修改 BMI .....	75
<b>6</b>	<b>编程工具（烧写工具） .....</b>	<b>76</b>
6.1	MemTool .....	76
6.1.1	Boot 模式 .....	76
6.1.2	烧写流程 .....	77
6.1.3	查看校验和 .....	83
6.1.4	读取 Flash 内容 .....	84
6.1.5	Flash 加密和解密 .....	87
6.2	XMC Flasher .....	91
6.2.1	烧写流程 .....	92
6.2.2	擦除芯片 .....	93
6.2.3	修改 BMI .....	93
<b>7</b>	<b>UC/PROBE .....</b>	<b>95</b>
7.1	uC/Probe 简介 .....	95
7.2	uC/Probe 使用说明 .....	95
7.3	uC/Probe 文件资源 .....	105
<b>8</b>	<b>开发套件 .....</b>	<b>106</b>
<b>9</b>	<b>XMC 相关链接 .....</b>	<b>110</b>

## 1. 快速入门

英飞凌推出基于 ARM Cortex-M0/4 的 XMC 系列 MCU 已经有近 4 年的时间了，在这期间，我们组织过很多 XMC MCU 培训和研讨会，也解决了很多客户实际项目中遇到的问题。但是在培训中，我们使用的大多是 PPT 格式的材料，这些材料一般只是列出结构框图和功能概要，而在解决客户问题的过程中，我们大多以邮件的方式讨论和解决问题。此书的编写目的就是要把以前的材料都整编在一起，给在使用 XMC MCU 的用户提供开发线索，从而让 XMC MCU 的开发和使用变得简单明了。

### XMC MCU 官方网站：

首先，英飞凌有一个很完善的官方网站，上面几乎能找到所有 XMC MCU 的开发资料，包括数据手册，工具，示例代码等。下面是英飞凌 XMC MCU 官网地址：

<http://www.infineon.com/xmc>

### XMC MCU 相关手册：

XMC MCU 的手册大致有 3 类：数据手册（Data Sheet），参考手册（Reference Manual）和勘误手册（Errata Sheet）。

数据手册主要包含 MCU 特性介绍，订购信息，封装类型，引脚定义，电气参数，外形尺寸等。

参考手册主要包括 CPU 子系统介绍（总线系统，指令集，通用寄存器），存储器结构和地址映射，时钟系统，外设介绍等。其中外设部分可分为 4 个部分，分别是通讯部分外设，模拟部分外设，工业控制部分外设，以及 GPIO 部分。

勘误手册主要记载异常功能（实际功能和参考手册不符的部分），异常的电气和时序（实测数据和数据手册中的数据不符的部分），以及实际应用中需要注意的一些点。用户在使用芯片的过程中如果怀疑可能是芯片出问题了，可以首先查一下勘误手册中有没有列出这一条。

除了这三类基础手册，我们还提供各个外设的应用笔记（Application Notes）以及附带的样例代码，帮助用户快速的掌握各个外设的基本功能。另外在 MCU 培训中使用的 PPT，也已经放在官网上了。

下面是 XMC1000 系列手册的下载地址：

<http://www.infineon.com/xmc1000>



Products

Highlights

**Documents**

Boards

Forums

Support

Software & Tools

## 32-bit XMC1000 Industrial Microcontroller ARM® Cortex®-M0

XMC1000 microcontrollers bring together the ARM® Cortex®-M0 core and market proven and differentiating peripherals in a leading-edge 65nm manufacturing process. XMC1000 is the number one choice to bring traditional 8-bit designs to the next level.

[Go to productlist](#)

NEW! XMC1400 Microcontroller series  
[Find products](#)

XMC™ Brochure  
[Download](#)

ARM® Cortex®-M0  
Core up to 48 MHz/peripherals up to 96 MHz

>70% performance increase

XMC1400  
up to 200 kB Flash

下面是 XMC4000 系列手册的下载地址：

<http://www.infineon.com/xmc4000>

Products	<h2>32-Bit XMC4000 Industrial Microcontroller ARM<sup>®</sup> Cortex<sup>®</sup>-M4</h2> <p>XMC4000 Families: XMC4800, XMC4700, XMC4500, XMC4400, XMC4300, XMC4200, XMC4100</p> <p>All XMC4000 devices are powered by ARM<sup>®</sup> Cortex<sup>®</sup>-M4 with a built-in DSP instruction set. The Single Precision Floating Point Unit, Direct Memory Access (DMA) feature and Memory Protection Unit (MPU) are state-of-the-art for all devices – even the smallest XMC4000 runs with up to 80MHz in core and peripherals. It comes with comprehensive set of common, fast and precise analog/mixed signal, Timer/PWM and communication peripherals.</p> <p>New XMC4300 series! &gt; <a href="#">Learn more</a></p> <p>XMC<sup>™</sup> Brochure <a href="#">Download</a></p> <p>As a highlight, the XMC4300 and XMC4800 devices are the first-ever highly integrated ARM<sup>®</sup> Cortex-M<sup>®</sup> based</p>
Highlights	
<b>Documents</b>	
Boards	
Forums	
Support	
Software & Tools	

### 开发环境：

DAVE4 是英飞凌自主开发的基于 Eclipse 的免费集成开发环境，它不仅包含 GNU 编译器和调试器，而且还集成了丰富的 APP。下面是 DAVE4 的下载地址：

<http://www.infineon.com/dave>

<b>DAVE<sup>™</sup></b>	 Download	Free Eclipse based integrated development environment (IDE) including GNU C-compiler, debugger, comprehensive code repository, hardware resource management, and code generation plug-in. <i>A complete download package is provided, including IDE, XMC<sup>™</sup> Lib, DAVE<sup>™</sup> APPs, EXAMPLES, and DAVE<sup>™</sup> SDK.</i> <a href="#">DAVE<sup>™</sup> Release Note</a>
<b>DAVE<sup>™</sup> SDK</b>		Development environment to modify and enhance existing DAVE <sup>™</sup> APPs and create new ones. DAVE <sup>™</sup> SDK is available as separate tool and installer. <a href="#">Release Note</a> All DAVE <sup>™</sup> APPs can be downloaded as individual DAVE <sup>™</sup> SDK projects.

### 开发工具（仿真器）：

XMC MCU 的仿真器主要有 J-Link 和 DAP miniWiddler. 英飞凌提供的开发套件一般都集成了 J-link. 另外英飞凌也有自己开发的 J-link (XMC-Link), 客户自己做的板子也可以使用标准的 Segger J-link. DAP miniWiddler 是英飞凌自己生产的 Jtag 调试器，除了调试 XMC MCU，也可以调试英飞凌其它私有核的 MCU。

XMC-Link 的介绍链接如下：

[www.infineon.com/xmclink](http://www.infineon.com/xmclink)

DAP MiniWiddler 的介绍链接如下：

<http://www.infineon.com/cms/en/product/microcontroller/development-tools-software-and-kits/dap-miniwiggler/channel.html?channel=db3a30432239cccd0122e537b4dc3c53>

## 编程 (烧写) 工具:

英飞凌原厂提供的烧写工具有 MemTool 和 XMC Flasher. MemTool 支持串口烧写 XMC MCU. XMC Flasher 支持 SWD 接口烧写 XMC MCU.

它们的下载链接如下:

[www.infineon.com/xmc/ecosystem](http://www.infineon.com/xmc/ecosystem)

# XMC Development Tools - Software and Tool Ecosystem

Following an overview of the XMC MCU Software and Tools Ecosystem offered by below listed third parties.

XMC Development Tools - Software and Tool Ecosystem subcategories

> Debug and Test Tools

> Programmers/Flash Tools

> Compiler Tool Chains

> RTOS and Middleware

> Training, Consulting and Design

> Software Downloads

> IEC60730 Class B

另外 XELTEK 和 WAVE 的离线烧写器支持对 XMC MCU 离线烧写。

关于 XELTEK 的烧写器可以访问下列网址:

<http://www.xeltek.com.cn/en/index.html>

关于 WAVE 的烧写器可以访问下列网址:

<http://www.wave-cn.com/>

## uc/Probe:

英飞凌最近发布了一款新的调试工具 – uc-Probe, 可以实时的观察变量或者外设寄存器的变化。它以各种控件的形式表示实际的量值, 例如文本框, 仪表盘, 甚至示波器。这种形式的调试会对一些需要动态观测参数的应用带来很大的方便, 例如电机和数字电源应用。但是使用它之前, 需要首先 download 程序到 MCU, 然后 load elf 文件到 uc-Probe。

uc-Probe 这个工具是 Micrium 开发的。一般来说, 开发者如果需要使用, 需向 Micrium 购买 License. 我们提供的版本是 Micrium 给 Infineon 定制的, 即只能调试 XMC MCU, 并且是免费提供给 XMC 用户使用的。它的下载链接如下:

<http://www.infineon.com/dave>

		<p>Graphically visualizes the internals of XMC™ 32-bit ARM® based microcontrollers</p> <p>The free-of-charge version <math>\mu</math>C/Probe™ XMC™ is a derivative of <math>\mu</math>C/Probe™ developed by Micrium® working with XMC™ microcontrollers only. <math>\mu</math>C/Probe™ XMC™ is a Windows</p>	
--	--	--	--

## 引脚分配工具:

XMC MCU 的管脚都是很多外设一起复用的，所以在分配引脚的时候给用户带来了一些困扰。因此我们开发了一个引脚分配工具（XMC Pinout Tool），可以根据所选择的外设来选择各个外设的功能引脚。这个工具在芯片选型和硬件设计的时候都是非常有用的。下面是这个工具的下载地址：

[www.infineon.com/xmc/ecosystem](http://www.infineon.com/xmc/ecosystem)

# XMC Development Tools - Software and Tool Ecosystem

Following an overview of the XMC MCU Software and Tools Ecosystem offered by below listed third parties.

## XMC Development Tools - Software and Tool Ecosystem subcategories

> Debug and Test Tools

> Programmers/Flash Tools

> Compiler Tool Chains

> RTOS and Middleware

> Training, Consulting and Design

> Software Downloads

> IEC60730 Class B

## XMC MCU 开发套件:

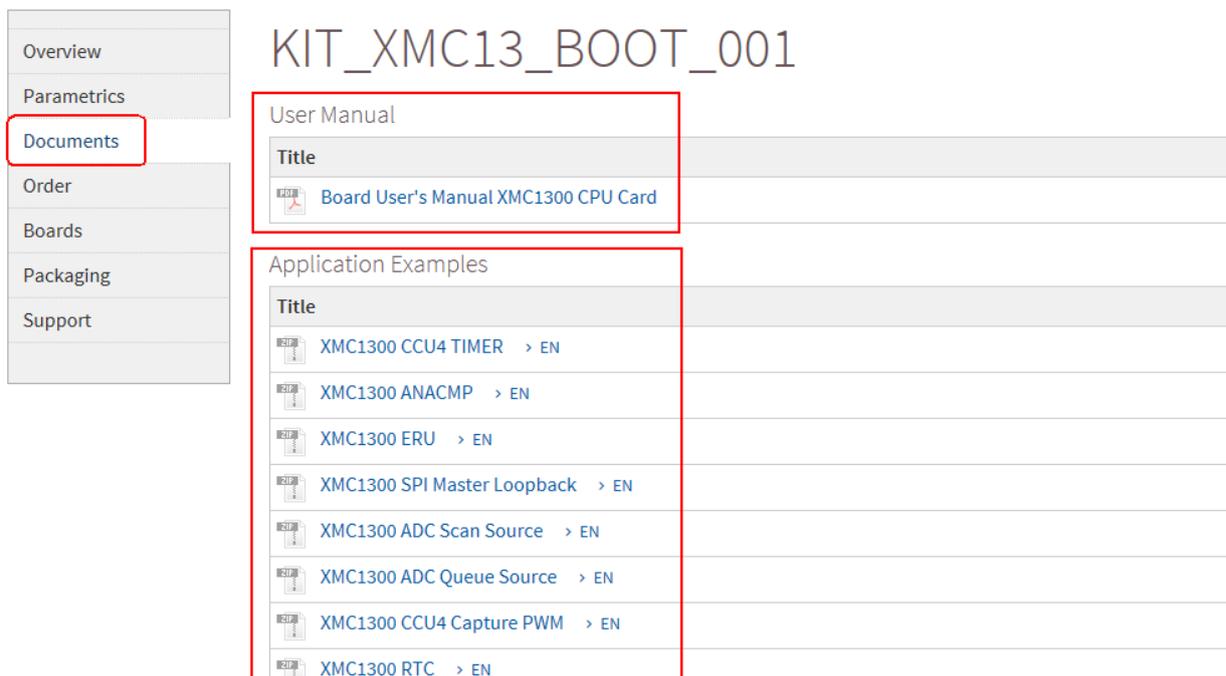
英飞凌提供很多 MCU 开发套件，可以在下列网站了解详细信息和在线购买:

[www.infineon.com/xmc-dev](http://www.infineon.com/xmc-dev)

对于每个开发套件，都能在各自链接找到评估板使用说明书（含原理图），附带例程以及 PCB 封装库等。

以 XMC1300 的开发套件为例:

[Products](#) > [Evaluation Boards](#) > [KIT\\_XMC13\\_BOOT\\_001](#)

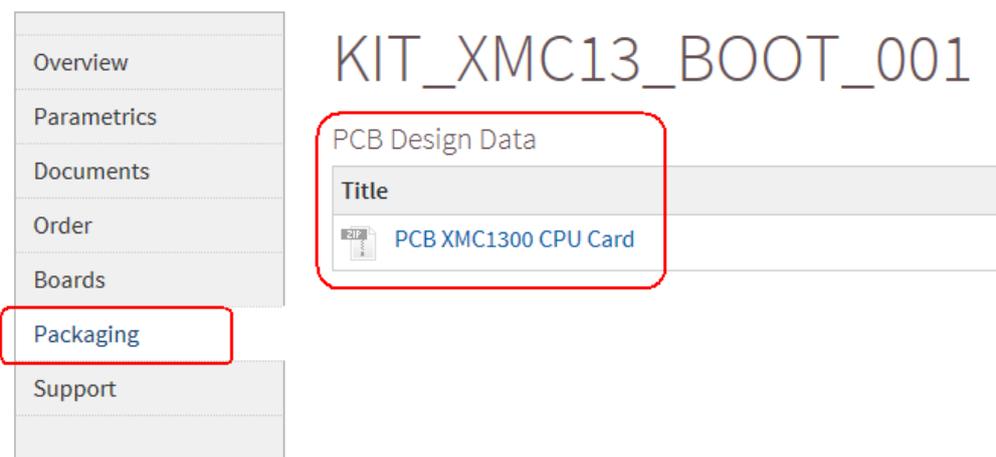


User Manual	
Title	
 Board User's Manual XMC1300 CPU Card	

Application Examples	
Title	
 XMC1300 CCU4 TIMER > EN	
 XMC1300 ANACMP > EN	
 XMC1300 ERU > EN	
 XMC1300 SPI Master Loopback > EN	
 XMC1300 ADC Scan Source > EN	
 XMC1300 ADC Queue Source > EN	
 XMC1300 CCU4 Capture PWM > EN	
 XMC1300 RTC > EN	

[Products](#) > [Evaluation Boards](#) > [KIT\\_XMC13\\_BOOT\\_001](#)



PCB Design Data	
Title	
 PCB XMC1300 CPU Card	

## 底层驱动库（Low Level Driver, LLD）：

英飞凌提供完整的底层驱动库，支持所有 XMC MCU，包含所有外设驱动。底层驱动库可以在下面链接下载。

[www.infineon.com/xmclib](http://www.infineon.com/xmclib)

DAVE™	 Download	Free Eclipse based integrated development environment (IDE) including GNU C-compiler, debugger, comprehensive code repository, hardware resource management, and code generation plug-in. <i>A complete download package is provided, including IDE, XMC™ Lib, DAVE™ APPs, EXAMPLES, and DAVE™ SDK.</i> <a href="#">DAVE™ Release Note</a>
DAVE™ SDK		Development environment to modify and enhance existing DAVE™ APPs and create new ones. DAVE™ SDK is available as separate tool and installer. <a href="#">Release Note</a> All DAVE™ APPs can be downloaded as individual DAVE™ SDK projects.
XMC™ Lib	 Download	Ready to use APIs for peripherals which are tested for GNU-, ARM-, IAR- und TASKING- compiler, and released for Altium, ARM/KEIL, Atollic, IAR Systems, iSystems and Rowley compiler IDEs. Low level driver libraries for XMC™ peripherals (APIs), CMSIS / MISRA 2004 compliant including documentation. <a href="#">XMC™ Lib – Release Note</a>

DAVE4 中已经把底层驱动库整合进去，所有 DAVE4 中的例程和 APP 都是基于底层驱动库编写的。

IAR Embedded Workbench v7.x 和 Keil MDK v5.x 也已经把底层驱动库整合进去，后续章节会介绍如何在 IAR 和 Keil 环境下使用底层驱动库。

## RTOS:

XMC MCU 已经支持很多 RTOS，包括 FreeRTOS, uC-OS, RTX 等。关于 XMC MCU 在这些 RTOS 移植的示例程序可以在下面链接下载：

[www.infineon.com/xmc/ecosystem](http://www.infineon.com/xmc/ecosystem)

## XMC Development Tools - Software and Tool Ecosystem

Following an overview of the XMC MCU Software and Tools Ecosystem offered by below listed third parties.

XMC Development Tools - Software and Tool Ecosystem subcategories

> Debug and Test Tools

> Training, Consulting and Design

> Programmers/Flash Tools

> Software Downloads

> Compiler Tool Chains

> IEC60730 Class B

> RTOS and Middleware

## 官方论坛：

英飞凌有一个技术社区，里面有关于 XMC MCU 的技术论坛，里面有很多技术问题的讨论，用户碰到问题可以放到那个论坛上，也有可能从那个论坛上找到问题的答案。另外社区中也有很多技术可以下载。下面这个是技术社区的连接：

<http://www.infineonic.org/>

除了中文的技术社区，英飞凌还有官方的英文论坛，在这个论坛，有新加坡和德国 XMC MCU 技术专家维护和支持，用户也可以在这个论坛上寻求支持。英文官方论坛地址如下：

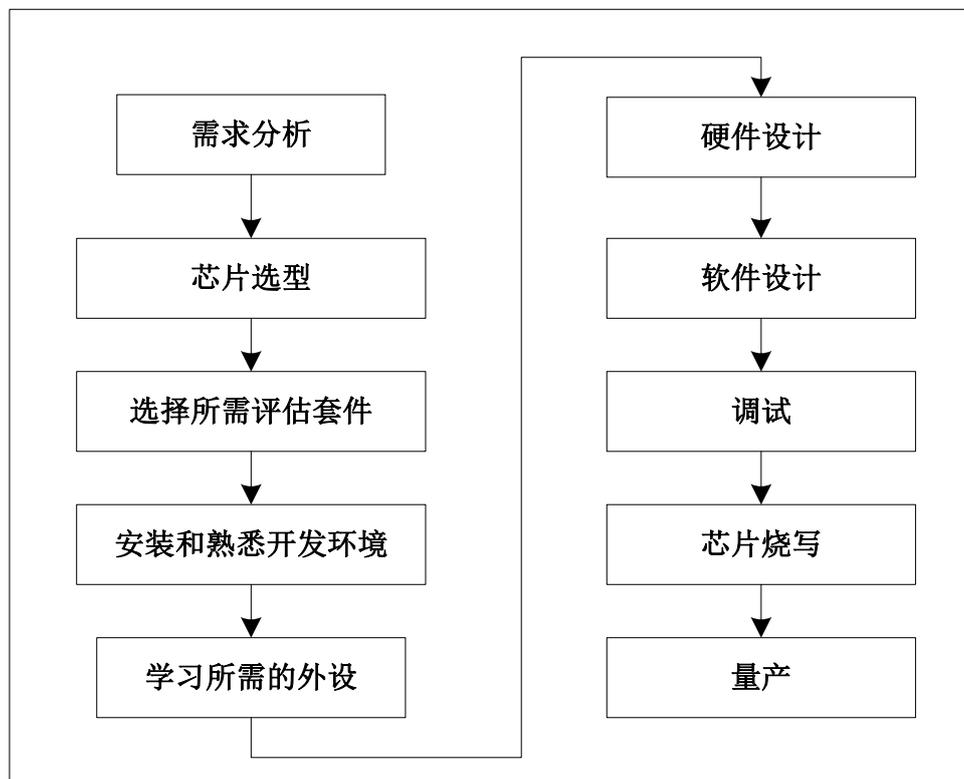
<http://www.infineonforums.com/>

## 2. 项目开发流程

### 2.1 流程概述

对于 XMC MCU 的新手来说，很多人都觉得使用英飞凌的 XMC MCU 去开发一个项目会无从下手。在这里，我们站在用户的立场上去思考怎么样让一个从来没使用过 XMC MCU 的新手能慢慢的熟悉 XMC MCU，并快速的将 XMC MCU 应用到实际项目中去。

一般基于 MCU 的项目的开发流程如下：



下面表中列出在开发流程中，英飞凌可以提供的一些帮助，从而加快项目的研发。

开发流程	英飞凌能提供什么	备注
需求分析	在英飞凌的官方网站可以下载到芯片数据手册和用户手册，帮助分析各个产品是否满足需求。	如需查看封装和外设资源，则可以查看数据手册。
芯片选型	在英飞凌的官方网站上就可以对芯片进行选型： <a href="http://www.infineon.com/xmc1000">http://www.infineon.com/xmc1000</a> <a href="http://www.infineon.com/xmc4000">http://www.infineon.com/xmc4000</a>	-

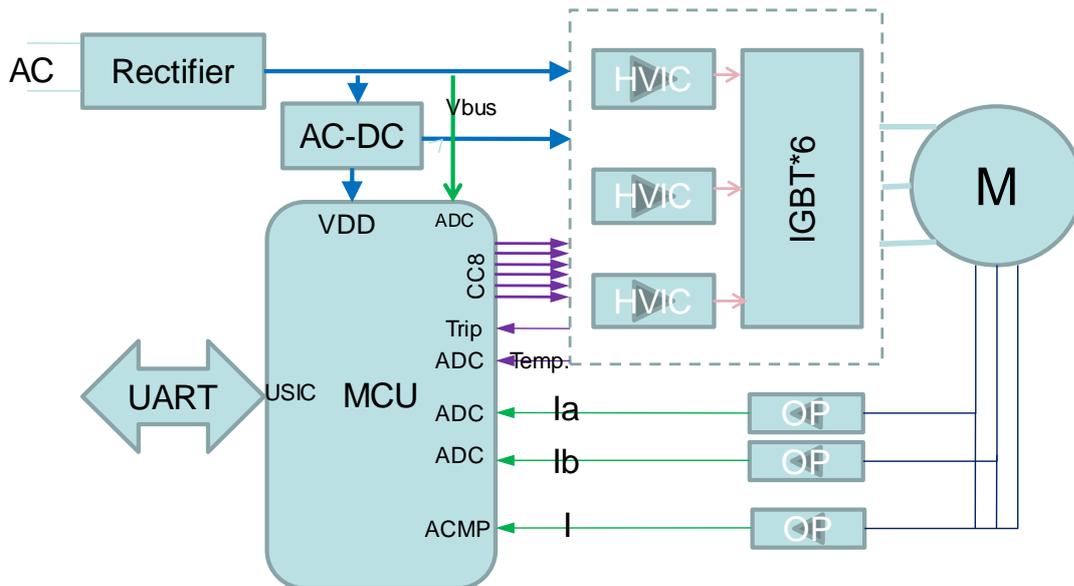
选择评估套件	<p>在下列网站可以选择和购买需要的评估套件，可以下载相关文档和示例程序： <a href="http://www.infineon.com/xmc-dev">www.infineon.com/xmc-dev</a></p>	<p>英飞凌提供基本的入门套件（Boot Kit），演示套件(Demo Kit)和应用套件(Applicaiton Kit)</p>
熟悉和安装开发环境	<p>DAVE下载地址（免费）： <a href="http://www.infineon.com/dave">http://www.infineon.com/dave</a></p> <p>Keil下载地址： <a href="http://www.keil.com">http://www.keil.com</a></p> <p>IAR下载地址： <a href="http://www.keil.com">http://www.keil.com</a></p>	<p>DAVE是基于Eclipse的免费开发环境，它内部集成了APP，可基于GUI配置直接生成代码。但是它的编译和执行效率较低。</p> <p>Keil MDK和IAR Embedded Workbench是专用的集成开发环境，在市场广泛使用，编译和执行效率都较高，但是需要购买Lisence才能使用。</p>
学习所示外设	<p>英飞凌提供3种形式的例程。</p> <ul style="list-style-type: none"> <li>- 基于LLD的例程</li> <li>- 直接寄存器访问的例程</li> <li>- 基于APP的例程</li> </ul>	<p>基于LLD的例程可以在英飞凌的官网上下载到。直接寄存器访问的例程可以在英飞凌的社区论坛上找到，APP 例程则集成在DAVE中。</p>
硬件设计	<p>因为XMC MCU的GPIO是复用的，一个GPIO通常有多种功能，因此英飞凌提供了引脚分配工具，在硬件设计的时候，可以帮助工程师进行引脚分配。 <a href="http://www.infineon.com/xmc/ecosystem">www.infineon.com/xmc/ecosystem</a></p>	<p>其实英飞凌还发布一个文档： APP_XMC4000_PCB_Design_Guidelines_v1_0 描述在用XMC4000时候，硬件设计上的一些注意事项。</p>
修改BMI	<p>在进行软件设计之前，如果使用XMC1000，则需要注意BMI的模式。新的XMC1000芯片出厂时的BMI是ASC_BSL模式，这个模式是用来串口烧写的。如果需要调试，则需要把BMI模式改成USER_MODE(SWD)。</p>	<p>目前提供了三种方法修改BMI： 在DAVE中可以修改BMI, 请查看<a href="#">4.4.6节</a> 使用J-Link命令行可以修改BMI，请查看<a href="#">5.4.2</a> 使用XMC Flasher可以修改BMI，请查看<a href="#">6.2.3</a></p>
软件设计和调试	<p>英飞凌最近发布了一款软件-uc/Probe,可以把变量和GUI控件联系起来，用控件的方式观察变量的实时变化。下载地址如下： <a href="http://www.infineon.com/dave">http://www.infineon.com/dave</a></p>	<p>uc-Probe这个工具是Micrium开发的，一般来说，开发者需要使用需向Micrium购买License. 我们提供的版本是Micrium给Infineon定制的，即只能调试XMC MCU，并且是免费提供给大家用的。</p>
芯片烧写	<p>烧写可以分为串口烧写和SWD接口烧写。</p> <p>下面工具支持串口烧写：</p> <ul style="list-style-type: none"> <li>- MemTool</li> <li>- 伟福J-Master烧写器</li> <li>- 希尔特烧写器</li> <li>-</li> </ul> <p>下面工具支持SWD接口烧写：</p> <ul style="list-style-type: none"> <li>- XMC Flasher</li> <li>- 伟福J-Master烧写器</li> <li>- XELTEK烧写器</li> </ul>	<p>在测试或者小批量生产时，可以使用MemTool和XMC Flasher烧写少量芯片。这两个工具的下链接如下： <a href="http://www.infineon.com/xmc/ecosystem">www.infineon.com/xmc/ecosystem</a></p> <p>在批量生产时，可以使用伟福J-Master烧写器或者XELTEK烧写器。</p> <p>关于XELTEK的烧写器可以访问下列网址： <a href="http://www.xeltek.com.cn/en/index.html">http://www.xeltek.com.cn/en/index.html</a></p> <p>关于WAVE的烧写器可以访问下列网址： <a href="http://www.wave-cn.com/">http://www.wave-cn.com/</a></p>

## 2.2 项目开发实例

下面以冰箱变频控制器为例，来介绍开发流程的各个步骤。

### 需求分析：

下图是冰箱变频控制器的结构框图，在这个系统中，MCU 需要用 3 对互补 PWM 去控制伺服电机，用 ADC 去采样电流 a 相和 b 相，用模拟比较器去比较 I 相电压，另外需要一路 UART 和主板通讯。



下面是基于系统分析后，对 MCU 的详细需求：

主频	$\geq 32\text{MHz}$
Flash	$\geq 64\text{KB}$
RAM	$\geq 8\text{KB}$
PWM 输出	3 对互补
ADC 通道	4
UART 通道	1
模拟比较器	1

## 芯片选型:

根据上面需求, 可以稍微做进一步的分析:

主频	$\geq 32\text{MHz}$	可以考虑 M0 的 32MHz 产品
Flash	$\geq 100\text{KB}$	至少要 128KB
RAM	$\geq 16\text{KB}$	
PWM 输出	3 对互补	需要 1 unit CCU8
ADC 通道	4	
UART 通道	1	需要 1 unit USIC
模拟比较器	1	需要有 ACMP

接下去可以打开下面链接进行芯片选型:

<http://www.infineon.com/xmc1000>

打开链接后往下拉, 看到下列页面后, 点击“Show all Parameter”, 然后它就会把所有的参数显示出来。

[Show all Parameters](#) > [Download Spreadsheet](#) > [Reset All Selections](#)

Product	OPN	Product Status	Order Online	Packages	Green	Budgetary Price €/1k	Clock Frequency	SRAM (incl. Cache)	CAN Nodes	A/D Input Lines (incl. FADC)
Matching Results: 137										
		active and preferred coming soon	Buy online	PG-LQFP-64 PG-TSSOP-16 PG-TSSOP-28 PG-TSSOP-38	yes	0.55 0.6 0.64 0.69	max 32.0 MHz 48.0 MHz	16.0 kByte	0 2 -	6 8 10 12

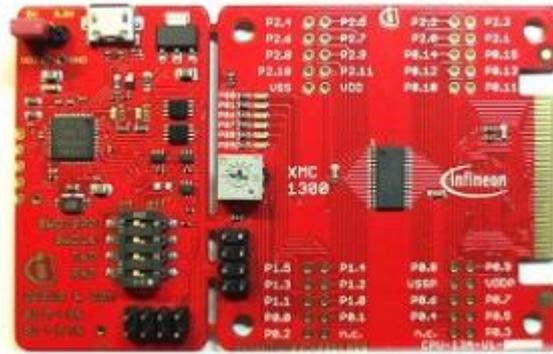
接着可以先选择 32MHz 的主频, 然后再选择 128KB 的 Flash size, 再接着选择 1 路 CCU8 和 USIC, 接着选择一个芯片的封装, 就可以找到 XMC1302-T038X0128AB 这个型号的 MCU。

[Show default Parameters](#) > [Download Spreadsheet](#) > [Reset All Selections](#)

Product	OPN	Product Status	Order Online	Packages	Green	Budgetary Price €/1k	Clock Frequency	SRAM (incl. Cache)	CAN Nodes	A/D Input Lines (incl. FADC)	Program Memory	Type of Memory	FlexRay™	Number of ADC Modules	Digital I/O Pins	EEPROM Emulati (Data-Flash)
Matching Results: 1																
		active and preferred		PG-TSSOP-38	yes	1.48	max 32.0 MHz	16.0 kByte	0	12	128.0 kByte	Flash	no	2.0	34	yes
XMC1300																
> XMC1302-T038X0128 AB		active and preferred		> PG-TSSOP-38		1.48	32.0 MHz	16.0 kByte	0	12	128.0 kByte	Flash	no	2.0	34	yes

### 选择所需评估套件:

对于从未使用过 XMC MCU 的新手，在开发项目之前，非常有必要选择一套评估套件进行学习，对于 XMC1300 系列可以选择下列套件（套件链接见第一章）：



### 安装和熟悉开发环境:

我们推荐使用的开发环境有 DAVE4, Keil MDK, IAR Embedded Workbench. DAVE4 是免费的，Keil MDK 和 IAR Embedded Workbench 则需要购买 license 或者只能使用限制版本（30 天或者 32KB 版本）。

DAVE4 的下载链接如下:

[www.infineon.com/dave](http://www.infineon.com/dave)

Keil 的下载，请访问 Keil 的官方网站:

<http://www.keil.com/>

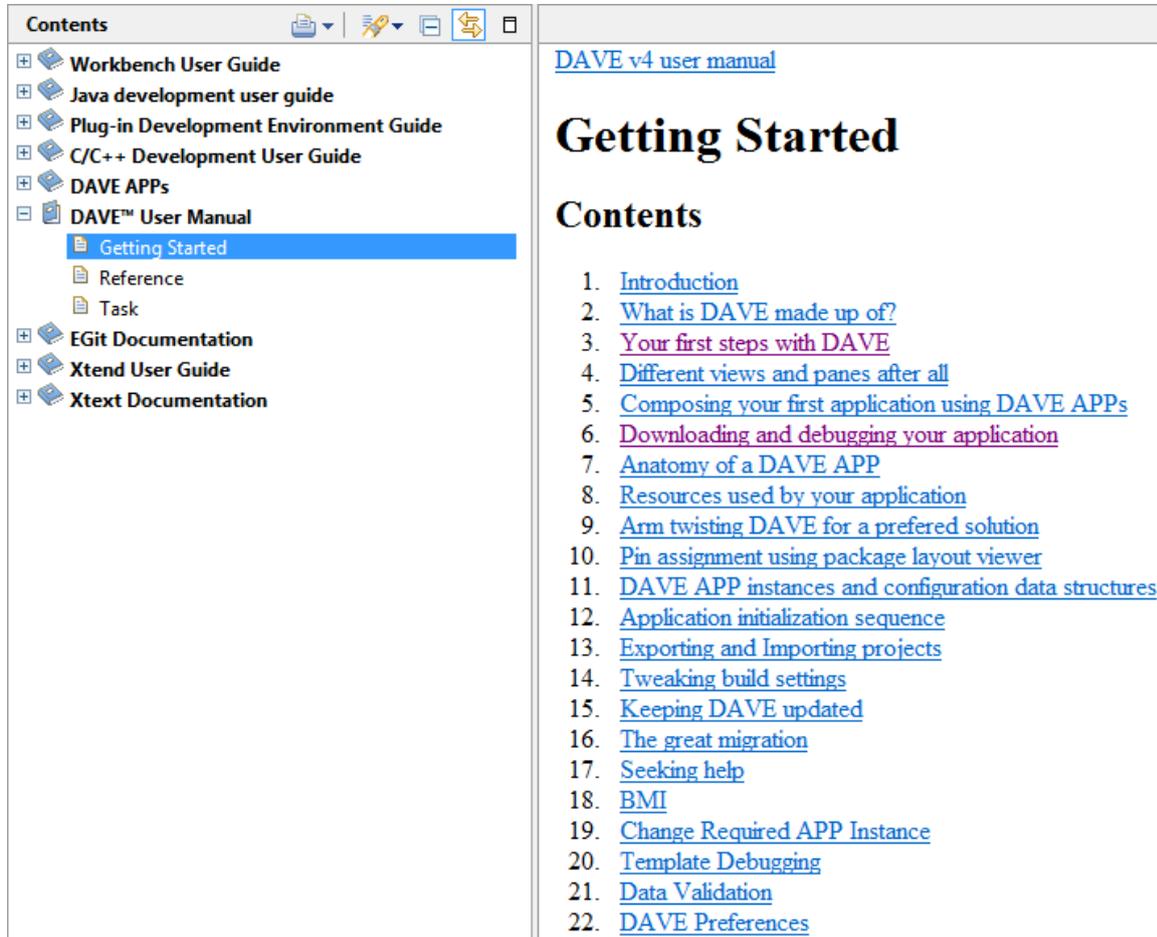
IAR 的下载，请访问 IAR 的官方网站:

<https://www.iar.com/>

现在的开发环境一般都集成非常强大的帮助文档（用户手册）。在阅读帮助文档后，用户可以非常快的熟悉开发环境。

点击工具条中的 **Help | Help Contents** 能打开 DAVE 的帮助文档:

点击 [DAVE User Manual | Getting Started](#)



**Contents**

- Workbench User Guide
- Java development user guide
- Plug-in Development Environment Guide
- C/C++ Development User Guide
- DAVE APPs
- DAVE™ User Manual
  - Getting Started**
  - Reference
  - Task
- EGit Documentation
- Xtend User Guide
- Xtext Documentation

[DAVE v4 user manual](#)

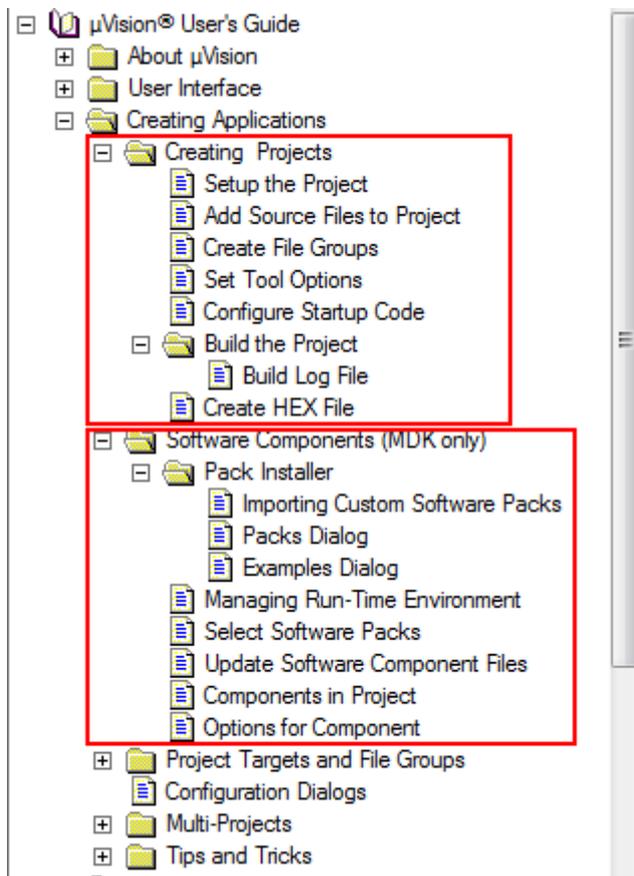
## Getting Started

### Contents

- [Introduction](#)
- [What is DAVE made up of?](#)
- [Your first steps with DAVE](#)
- [Different views and panes after all](#)
- [Composing your first application using DAVE APPs](#)
- [Downloading and debugging your application](#)
- [Anatomy of a DAVE APP](#)
- [Resources used by your application](#)
- [Arm twisting DAVE for a preferred solution](#)
- [Pin assignment using package layout viewer](#)
- [DAVE APP instances and configuration data structures](#)
- [Application initialization sequence](#)
- [Exporting and Importing projects](#)
- [Tweaking build settings](#)
- [Keeping DAVE updated](#)
- [The great migration](#)
- [Seeking help](#)
- [BMI](#)
- [Change Required APP Instance](#)
- [Template Debugging](#)
- [Data Validation](#)
- [DAVE Preferences](#)

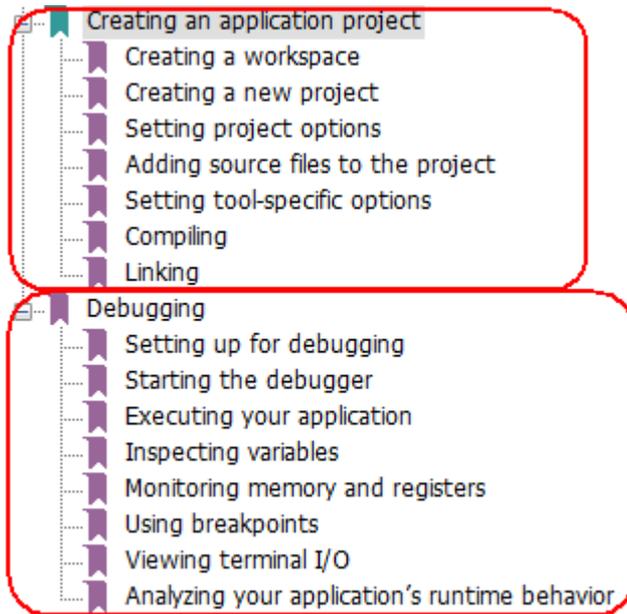
对于从未使用过 DAVE 的新手，可以先看 **Your First steps with DAVE**，里面介绍了如何新建一个工程，如何使用 DAVE APP 等，接着可以看 **Composing your first application using DAVE APPs**，里面会介绍如何把两个 APP 结合起来使用在一个项目中，接着可以看 **Downloading and debugging your application**，里面介绍如何下载和调试代码。学习完这三个步骤后就基本能够正常使用 DAVE4 这个环境了。

在 Keil MDK 中，点击 **Help | uVision**，可以打开 Keil 的帮助文档：



首先可以点击”**Creating Projects**”，就可以查看如何在 Keil 环境下新建一个工程，对于 Keil MDK 5 以上的版本，Pack 是非常有用的组件，里面集成了很多有用的软件，例如底层驱动库，CMSIS 库，中间件（RTOS，TCP/IP 等），例程等。可以查看 **Pack Installer**，可以了解怎么样下载和安装 Packs。

在 IAR Embedded Workbench 中，点击 **Help | Getting Started with IAR Embedded Workbench**，可以打开 IAR 的帮助文档。查看 “**Creating an application project**”，可以了解怎么样新建工程。查看 “**Debugging**”，了解怎么样调试。



### 学习所需外设:

XMC MCU 有非常丰富的外设资源，但是在开发时间有限的情况下，可以只学习项目中用到的一些外设。例如上面这个冰箱变频控制板这个项目，可以只学习 ADC, CCU8, UART, GPIO, ACMP 这几个外设。

我们提供几种形式的例程：基于 LLD 的例程，直接访问寄存器的例程，基于 APP 的例程，以及开发套件自带的一些例程。

对于**基于 LLD 的例程**，首先我们提供很多基于 DAVE4 LLD 例程，可以在下列网站下载：

下面是 XMC1000 系列 LLD 例程的下载地址：

<http://www.infineon.com/xmc1000>

下面是 XMC4000 系列 LLD 例程的下载地址：

<http://www.infineon.com/xmc4000>

- Products
- Highlights
- Documents
- Boards
- Forums
- Support
- Software & Tools

## 32-Bit XMC4000 Industrial Microcontroller ARM<sup>®</sup> Cortex<sup>®</sup>-M4

XMC4000 Families: XMC4800, XMC4700, XMC4500, XMC4400, XMC4300, XMC4200, XMC4100  
 All XMC4000 devices are powered by ARM<sup>®</sup> Cortex<sup>®</sup>-M4 with a built-in DSP instruction set. The Single Precision Floating Point Unit, Direct Memory Access (DMA) feature and Memory Protection Unit (MPU) are state-of-the-art for all devices – even the smallest XMC4000 runs with up to 80MHz in core and peripherals. It comes with comprehensive set of common, fast and precise analog/mixed signal, Timer/PWM and communication peripherals.

As a highlight, the XMC4300 and XMC4800 devices are the first-ever highly integrated ARM<sup>®</sup> Cortex-M<sup>®</sup> based

New XMC4300 series!  
[> Learn more](#)

XMC<sup>™</sup> Brochure  
[Download](#)

这些例程是和应用笔记相结合的，可以先看应用笔记再跑例程代码，这个能达到最好的学习效果。

在 Keil MDK v5.x 和 IAR Embedded Workbench v7.x 中，已经集成了 LLD 和很多示例代码，关于如何在 Keil 和 IAR 上使用 LLD，可以打开上面链接，查看下面文档：

<a href="#">Tooling - Import DAVE™ version 4 Generated Library Sources into Atollic TrueSTUDIO® for ARM® &gt; EN</a>	1 MB	08 May 2015	01_00
<a href="#">Tooling - Import DAVE™ version 4 Generated Library Sources into IAR Embedded Workbench® for ARM® &gt; EN</a>	924 KB	08 May 2015	01_00
<a href="#">Tooling - Import DAVE™ version 4 Generated Library Sources to ARM® MDK using CMSIS® PACK &gt; EN</a>	1.3 MB	08 May 2015	01_00
<a href="#">Tooling - Import DAVE™ version 4 Generated Library Sources into Rowley CrossWorks® for ARM® &gt; EN</a>	876 KB	08 May 2015	01_00
<a href="#">Tooling - Integrate XMC Lib LLD in 3rd Parties Tool Chains (Keil, IAR, Atollic and Rowley) &gt; EN</a>	1.4 MB	03 Jun 2015	01_01

对于直接访问寄存器的例程，放在英飞凌中文社区的资料下载里面：

首页
新闻
英飞凌汽车电子生态圈
技术文章
活动
论坛
视频
资料下载
关键字

文档 / 单片机

所有文档

- 汽车功率电子
- 工业功率电子
- 单片机
- XMC1000
- XMC4000
- XC800
- XC2000/XE166
- Tricore
- 传感器
- 风机水泵控制秀
- 研讨会资料

### 单片机

标题	中文文档	英文文档	附件	上传
<a href="#">ACInduction_Motor_GS_Example01</a>	无	无		201
<a href="#">ACInduction_Motor_GS_Example02</a>	无	无		201
<a href="#">ACInduction_Motor_xSPY_GS_Example01</a>	无	无		201
<a href="#">ADC</a>	无		无	201
<a href="#">ADC Gate&amp;Trigger</a>		无		201
<a href="#">ADC002_Example1</a>	无	无		201
<a href="#">ADC003_Example1</a>	无	无		201
<a href="#">ADC_CCU8周期中断触发4路AD并行采样</a>	无	无		201

例如搜索 ADC, 就能找到很多基于 Keil 或者 DAVE 的寄存器直接访问的例程。

[Keil\\_XMC1200\\_ADC\\_后台运行模式](#)

2014-12-18 17:49:50

[Keil\\_XMC1200\\_ADC\\_队列模式](#)

2014-12-18 17:49:13

[Keil\\_XMC1200\\_ADC\\_扫描模式](#)

2014-12-18 17:48:29

[Keil\\_XMC1300\\_ADC\\_后台运行模式](#)

2014-12-18 17:41:29

[Keil\\_XMC1300\\_ADC\\_队列模式](#)

2014-12-18 17:40:41

[Keil\\_XMC1300\\_ADC\\_扫描模式](#)

2014-12-18 17:39:47

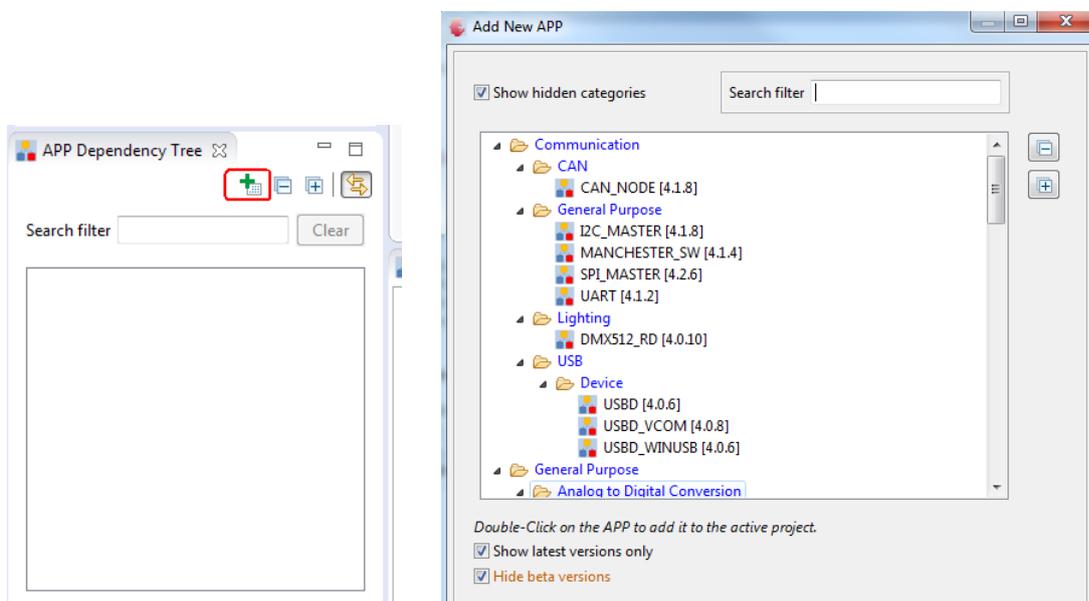
[Keil\\_XMC1300\\_ADC\\_多路同步转换例程](#)

2014-12-18 17:04:15

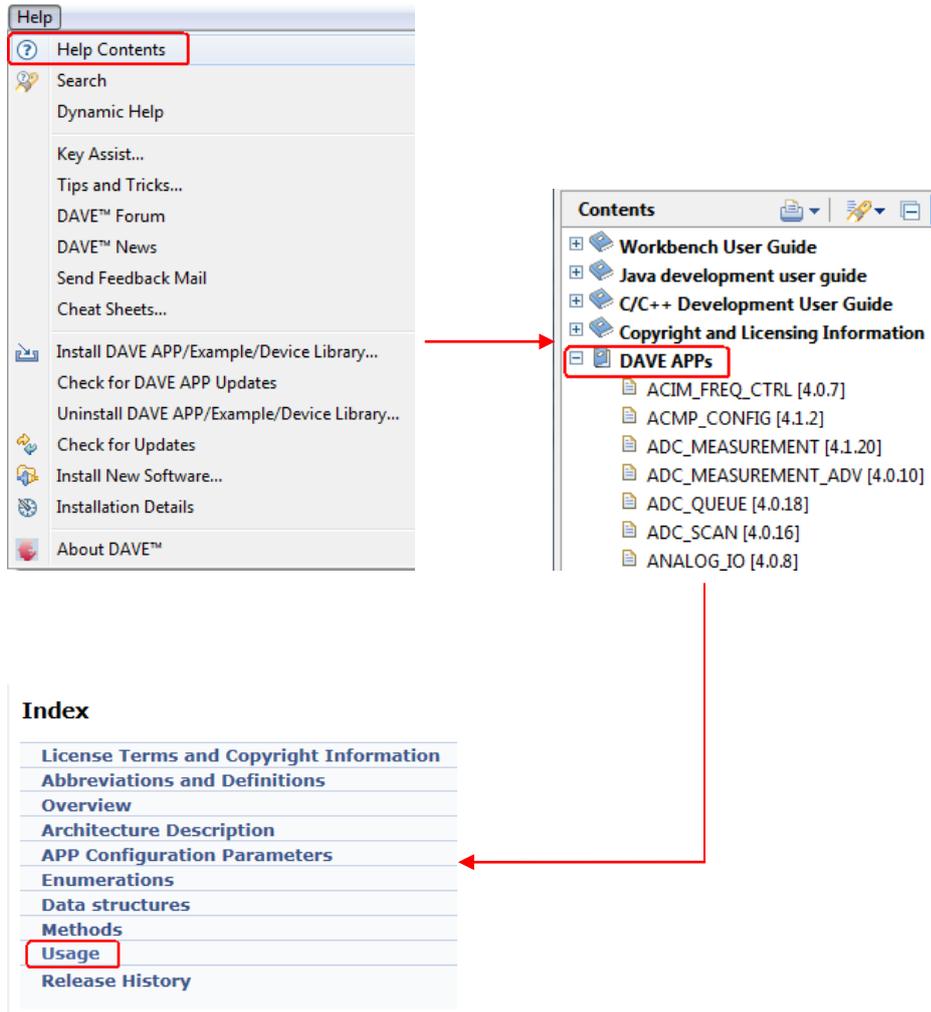
[Keil\\_XMC1300\\_ADC\\_扫描模式例程](#)

对于基于 **APP** 的例程, 可以打开 DAVE4, 然后新建一个 DAVE CE 的工程, 然后就能选择 APP, 把它导入到工程, 接着点击 **Generate Code**, 编译后就能使用这个 APP 了。

点击下面加号选择 APP:



对于如何使用这个 APP，则可以看说明文档，如下所示。

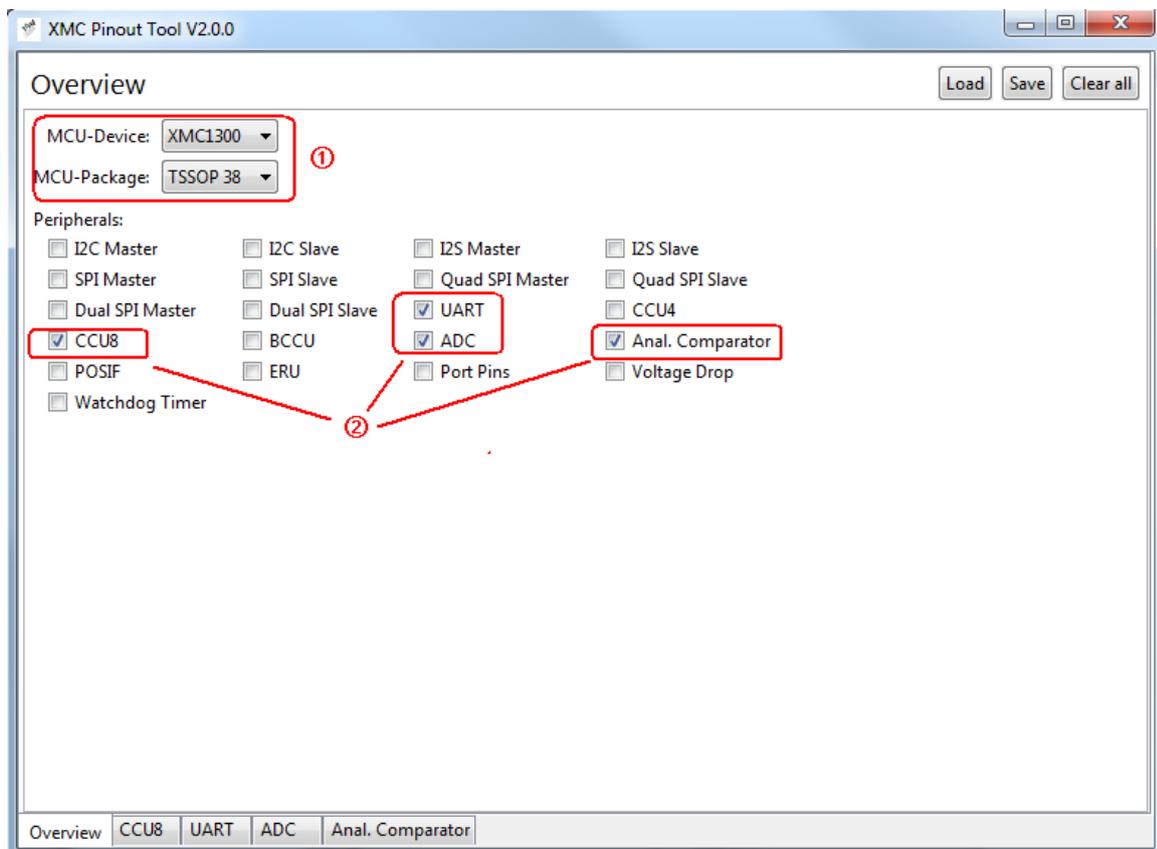


## 硬件设计：

在这个系统中，MCU 需要用 3 对互补 PWM 去控制伺服电机，用 ADC 两个通道去采样电流 a 相和 b 相，另两个通道去采样温度和供电电压，用模拟比较器去比较 I 相电压，另外需要一路 UART 和主板通讯。

我们可以使用引脚分配工具（XMC Pinout tool）进行引脚分配。这个工具的下载地址见第一章。

- 1) 打开工具后，先选择 MCU 型号和封装
- 2) 选择所需的外设



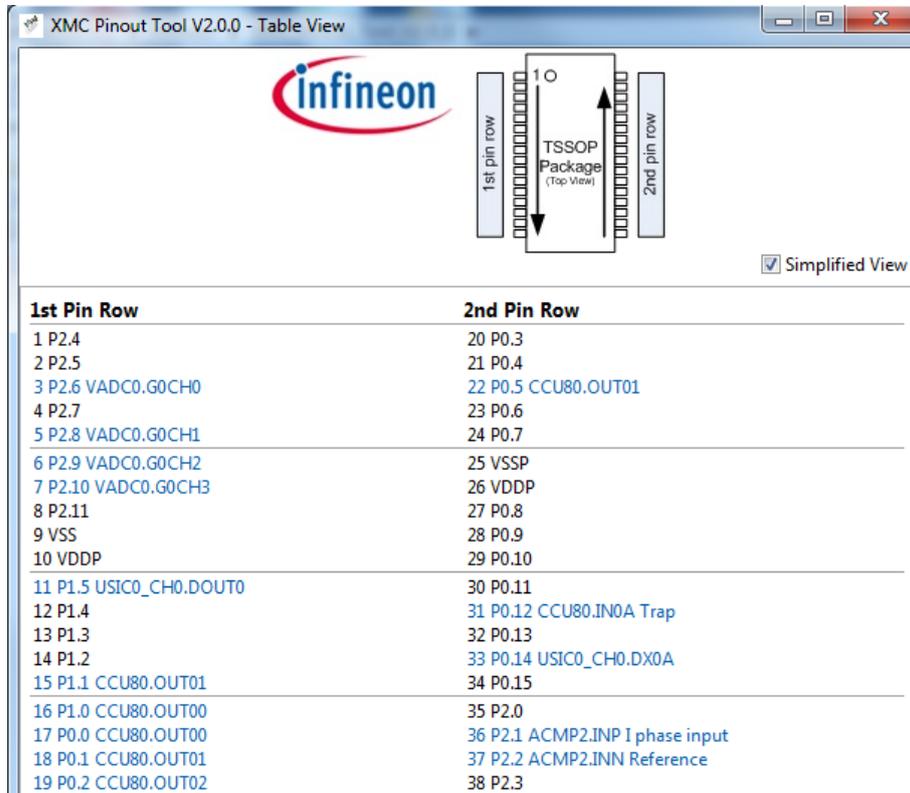
3) 接着就可以选择各个外设所要使用的引脚。

这里选择使用 CCU80 产生 3 对互补 PWM，USIC 的通道 0 作为 UART 模式，ADC 的 group0 的 4 个通道去进行采样，ADC 比较器的通道 2 去做模拟比较输入。

The image shows a configuration tool interface with several panels:

- CCU80:**
  - Slice 0 Input:**  P0.12, IN0A Trap
  - Slice 0 Output:**
    - P1.1, OUT01
    - P0.2, OUT02
    - P0.5, OUT01
    - P1.0, OUT00
    - P0.1, OUT01
    - P0.0, OUT00
  - Slice 1 Input:**
    - P0.12, IN1A
    - P0.5, IN1B
  - Slice 1 Output:**
    - P1.2, OUT10
    - P0.3, OUT11
    - P0.2, OUT10
- UART:**
  - P1.0, DX0C
  - P2.9, DX5A
  - P2.8, DX3D
  - P0.14, DX0A
  - P2.5, DX5D
  - P2.4, DX3B
  - P2.7, DX5C
  - P0.15, DX0B
  - P2.6, DX3E
  - P2.10, DX3C
  - TXD:**
    - P2.1, DOUT0
    - P1.1, DOUT0
    - P2.0, DOUT0
    - P1.0, DOUT0
    - P0.14, DOUT0
    - P1.5, DOUT0
    - P0.15, DOUT0
- ADC (VADC0):**
  - Group 0:**
    - P2.1, G0CH6
    - P2.0, G0CH5
    - P2.11, G0CH4
    - P2.2, G0CH7
    - P2.10, G0CH3
    - P2.9, G0CH2
    - P2.8, G0CH1
    - P2.6, G0CH0
- Anal. Comparator:**
  - ACMP:**
    - Reference for all ACMP:  P2.11, REF
    - Input: --- NONE ---
    - Output: --- NONE ---
  - ACMP0:**
    - Reference for all ACMP: --- NONE ---
    - Input:
      - P2.9, INP
      - P2.8, INN
    - Output:
      - P0.10, OUT
      - P2.10, OUT
  - ACMP1:**
    - Reference for all ACMP: --- NONE ---
    - Input:
      - P2.7, INP
      - P2.6, INN
    - Output:
      - P1.0, OUT
  - ACMP2:**
    - Reference for all ACMP: --- NONE ---
    - Input:
      - P2.1, INP I phase input
      - P2.2, INN Reference
    - Output:
      - P1.2, OUT
      - P0.5, OUT

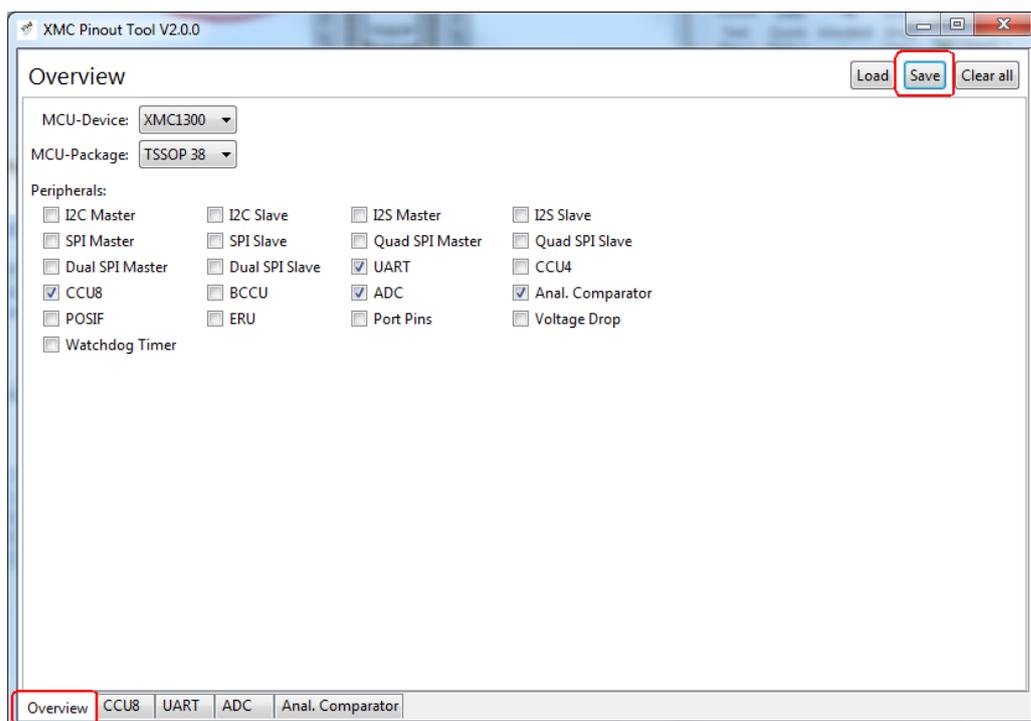
- 4) 接着就可以在另外一个串口看到如下引脚分配图，在画原理图的时候，可以直接根据这个表来分配引脚。



The screenshot shows the XMC Pinout Tool V2.0.0 interface. At the top, there is a diagram of a TSSOP Package (Top View) with two pin rows labeled '1st pin row' and '2nd pin row'. Below the diagram is a table with two columns: '1st Pin Row' and '2nd Pin Row'. The table lists pin numbers and their corresponding peripheral functions.

1st Pin Row	2nd Pin Row
1 P2.4	20 P0.3
2 P2.5	21 P0.4
3 P2.6 VADC0.G0CH0	22 P0.5 CCU80.OUT01
4 P2.7	23 P0.6
5 P2.8 VADC0.G0CH1	24 P0.7
6 P2.9 VADC0.G0CH2	25 VSSP
7 P2.10 VADC0.G0CH3	26 VDDP
8 P2.11	27 P0.8
9 VSS	28 P0.9
10 VDDP	29 P0.10
11 P1.5 USIC0_CH0.DOUT0	30 P0.11
12 P1.4	31 P0.12 CCU80.IN0A Trap
13 P1.3	32 P0.13
14 P1.2	33 P0.14 USIC0_CH0.DX0A
15 P1.1 CCU80.OUT01	34 P0.15
16 P1.0 CCU80.OUT00	35 P2.0
17 P0.0 CCU80.OUT00	36 P2.1 ACMP2.INP I phase input
18 P0.1 CCU80.OUT01	37 P2.2 ACMP2.INN Reference
19 P0.2 CCU80.OUT02	38 P2.3

- 5) 选择 Overview 的 table, 点击 Save, 也可以把引脚分配表保存成 excel 文档。



## 软件设计和调试:

根据前面介绍, 我们提供的例程有 3 种: 基于 LLD 的例程, 直接访问寄存器的例程和基于 APP 的例程。

如果使用的是基于 LLD 的例程, 那么在实际系统中, 可以根据项目需要, 把几个用到的外设封装成 High Level Driver, 例如 `uart.c`, `adc.c`, `ccu8.c`, 如果 UART 只是用到了查询模式, 那么 `uart.c` 中可以只实现 UART 查询模式的读写数据。

如果使用的是直接访问寄存器的例程, 那么在实际系统中, 可以根据项目需要, 把几个用到的外设封装成 Low Level Driver, 例如 `uart.c`, `adc.c`, `ccu8.c` 等。

如果使用的是基于 APP 的例程, 可以根据项目需要, 把几个 APP 整合在一起, 如何整合可以看 DAVE4 的帮助文档。关于调试, 也可以看各个开发环境的帮助文档。

另外, 英飞凌最近发布了一款新的调试工具 – `uc-Probe`, 可以实时的观察变量或者外设寄存器的变化。它以各种控件的形式表示实际的量值, 例如文本框, 仪表盘, 甚至示波器。这种形式的调试会对一些需要动态观测参数的应用带来很大的方便, 例如电机和数字电源应用。但是使用它之前, 需要首先 `download` 程序到 MCU, 然后 `load elf` 文件到 `uc-Probe`。具体请看 [第 7 章](#)。

## 芯片烧写:

芯片的烧写可以选择 XMC Flasher, MemTool, Xeltek 烧写器, WAVE 烧写器。

XMC Flasher 支持用 SWD 接口烧写 XMC MCU, MemTool 支持串口烧写 XMC MCU。它们的下载地址如下:

[www.infineon.com/xmc/ecosystem](http://www.infineon.com/xmc/ecosystem)

# XMC Development Tools - Software and Tool Ecosystem

Following an overview of the XMC MCU Software and Tools Ecosystem offered by below listed third parties.

XMC Development Tools - Software and Tool Ecosystem subcategories

> Debug and Test Tools

> **Programmings/Flash Tools**

> Compiler Tool Chains

> RTOS and Middleware

> Training, Consulting and Design

> Software Downloads

> IEC60730 Class B

另外 XELTEK 和 WAVE 的离线烧写器支持对 XMC MCU 离线烧写。

关于 XELTEK 的烧写器可以访问下列网址:

<http://www.xeltek.com.cn/en/index.html>

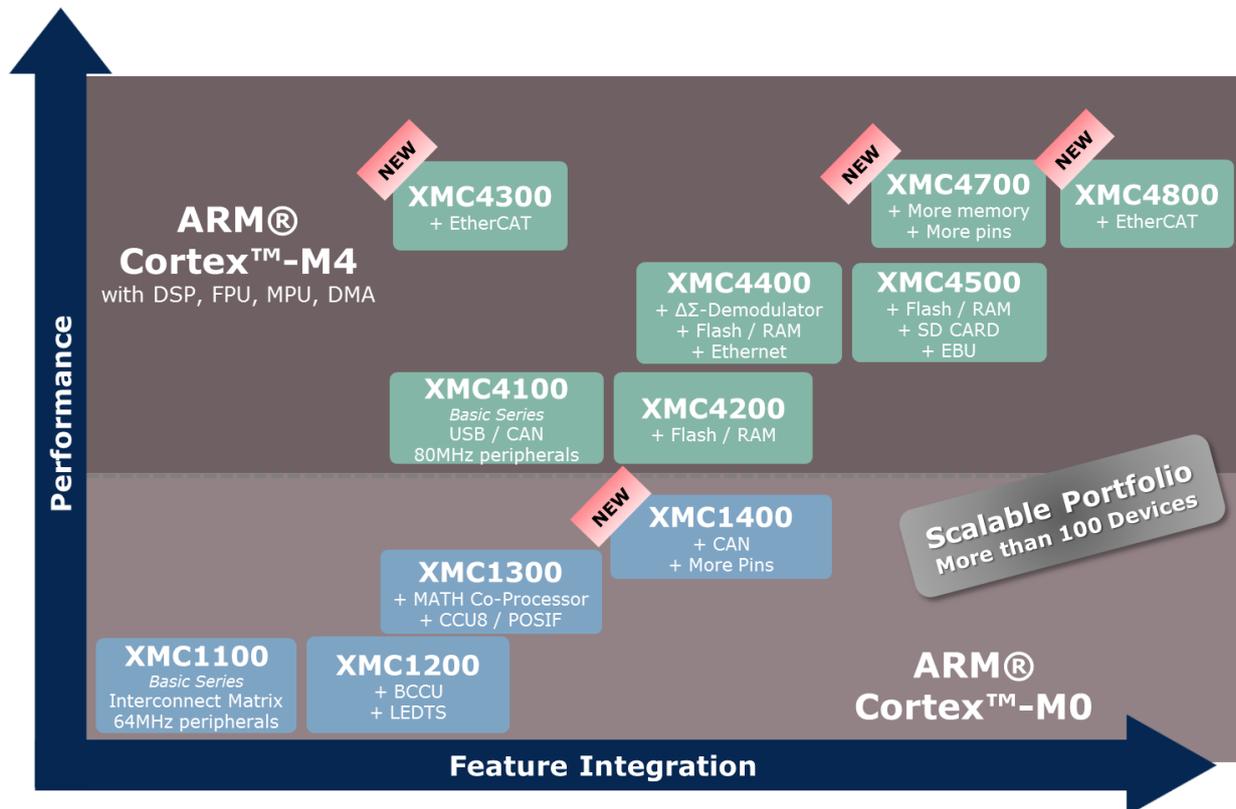
关于 WAVE 的烧写器可以访问下列网址:

<http://www.wave-cn.com/>

### 3. 产品介绍

#### 3.1 产品路线

XMC MCU 产品包括基于 ARM Cortex-M0 内核的 XMC1000 家族和基于 ARM Cortex-M4 内核的 XMC4000 家族。



XMC1000 家族目前有 XMC1100, XMC1200, XMC1300 和 XMC1400 四个系列的产品。XMC1100 系列是 XMC1000 家族中入门的产品，它有 32 位机的性能，但是价格和 8 位机不相上下，因此主要用来替换原来的 8 位机，用于一些简单的通用领域。

XMC1200 系列比 XMC1100 增加了 BCCU 单元，可实现硬件自动调光，非常适合 LED 调光的应用。

XMC1300 系列则在 XMC1200 的基础上又增加了 CCU8, MATH, POSIF 单元，非常适合于电机控制和数字电源的应用。一个 CCU8 单元可以输出 8 对互补 PWM 信号。MATH 单元可以用硬件实现三角函数，矢量运算以及除法运算。POSIF 是位置检测单元，可以检测 Hall 信号和位置编码器信号。

XMC1400 系列是最新推出的产品。和 XMC1300 相比，XMC1400 的主频提高到 48MHz, 并且增加了 CAN 模块。

XMC4000 家族目前有 XMC4100, XMC4200, XMC4300, XMC4400, XMC4500, XMC4700 和 XMC4800 七个系列的产品。

XMC4100 系列的 XMC4000 家族的入门级产品，主频达 80MHz，内部集成了很多用于电机控制的外设（CCU4, CCU8, POSIF）以及高精度 PWM 单元，可应用于白色家电（例如空调变频控制板）和需要高性能的数字电源领域。

XMC4200 是 XMC4100 的升级版，它和 XMC4100 是 pin to pin 兼容的，但是 RAM 和 Flash 比 XMC4100 都要大一倍。

XMC4400 系列与 XMC4200 和 XMC4100 相比，增加了 Delta-Sigma Demodulator (DSD) 和 Ethernet 模块，它的主频达到了 120MHz，因此可应用于高性能的通用变频器。

XMC4500 系列是 XMC4400 的升级版，它和 XMC4400 也是 pin to pin 兼容的，但是 RAM 和 Flash 比 XMC4400 都要大一倍，另外 XMC4500 增加了 SDMMC 模块。

XMC4700 是 XMC4500 的升级版，频率达到 144MHz，Ethernet 的通道上升到 2 个。

XMC4800 和 XMC4700 是 pin to pin 兼容，它集成了 EtherCAT 模块，可用于工业机器人的控制以及步进或伺服电机控制。

XMC4300 是 XMC4800 的简化版，去掉了 XMC4800 中大部分电机控制相关的外设，可用户 EtherCAT 的扩展 IO 模块或者通讯模块。

## 3.2 XMC1000 系列

### 3.2.1 产品特性

- 业界领先的 **65nm** 嵌入式闪存技术
- 32-bit **ARM Cortex™-M0**, 32/48MHz
- **64MHz MATH** 协处理器 (CORDIC / DIVIDE)
- Flash:**8~200K** 字节, RAM: **16K** 字节
- 工作电压范围: **1.8V – 5.5V**
- 用于 IP 加密的 **AES 128 位**加密下载模式
- **LED 亮度&色彩控制单元**
- 丰富的电机专用单元(CCU4,CCU8,POSIF,CMP)
- **兼容 IEC 60730 Class B**
- 电容式触控 & LED 显示

### 3.2.2 产品阵营

		Low-end			High-end
		XMC1100	XMC1200	XMC1300	XMC1400
System Performance	Core	ARM® Cortex™-M0			
	CPU frequency (at 105 °C)	32MHz			48MHz
	Math Processor			✓	✓
	Flash size	8-64 kB	16-200 kB	8-200 kB	32-200 kB
	RAM size	16 kB			
Industrial Control	POSIF			1x	2x
	CCU4 (4ch)	1x	1x	1x	2x
	CCU8 (4ch)			1x	2x
	BCCU (3ch)		1x		1x
Analog	ADC 12-bit	2x (1x S&H)	2x (2x S&H)	2x (2x S&H)	2x (2x S&H)
	Analog Comparator		2x	3x	4x
COM	USIC(UART, SPI, I <sup>2</sup> C, I <sup>2</sup> S)	2x	2x	2x	4x
	CAN				2x
	Touch Button	✓	✓	✓	✓

注：上面所写外设部分的通道数是最大值，其中各系列中又有很多 Part Number，每个 Part Number 的外设通道数和存储器容量都不一样，具体请看芯片的数据手册。

### 3.2.3 产品应用领域

- 电机控制（支持简单控制至无传感器 FOC）
  - 电动自行车，风机，电动工具
- LED 照明
  - HD LED, RGB LED 调光
  - DALI 和 DMX 协议
- 数字电源转换
  - PFC, LLC, 逆变器
- 传感器控制
- HMI (电容式触控 & LED 显示)
- 通用应用

### 3.3 XMC4000 系列

#### 3.3.1 产品特性

- ARM Cortex™-M4 , 内置 FPU (浮点运算单元)
- 主频: 80 -144MHZ
- 最快 Flash 访问,64KB-2.5MB,带 Cache, ECC
- RAM : 20KB-512KB
- 硬件 MPU,CRC
- 多通道 DMA
- 连接内部外设信号的 ERU (事件处理单元)
- 最强的 PWM 单元 (CCU4, CCU8)
- 多核多通道 12-Bit ADC
- 2 通道 12-Bit DAC
- 位置接口单元 (正交编码/Hall/多通道)
- 丰富的通讯接口 (USB,CAN,UART, SPI, I2C, I2S, Ethernet, EtherCAT)
- 丰富的人机接口 (Capacitive Touch, LED Matrix, Programmable Ports)
- 高分辨率 PWM 单元 (150ps)
- Delta/Sigma 解调器
- 业界唯一工作温度范围达 125°C 的 Cortex™-M4 产品

### 3.3.2 产品阵营

		Low-end					High-end
		XMC4100	XMC4200	XMC4400	XMC4500	XMC4700	XMC4800
System Performance	Core	ARM® Cortex™-M4					
	CPU frequency (at 125 °C)	80 MHz	80 MHz	120 MHz	120 MHz	144MHz	144MHz
	Co-proc	Floating Point Unit					
	Flash size	128 kB	256 kB	512 kB	1 MB	2.5 MB	2.5MB
	RAM size	20 kB	40 kB	80 kB	160 kB	512 kB	512 kB
	Cache	4 kB	4 kB	4 kB	4 kB	6 kB	6 kB
	Timers	POSIF	2x	2x	2x	2x	2x
CCU4 (4ch)		2x	2x	4x	4x	4x	4x
CCU8 (4ch)		1x	1x	2x	2x	2x	2x
High-resolution PWM (150ps)		4x	4x	4x			
Signal Processing	ADC 12-bit	2x	2x	4x	4x	4x	4x
	Delta/Sigma Demodulator			4x	4x	4x	4x
	DAC	2x	2x	2x	2x	2x	2x
Communication	Ethernet MAC/EtherCAT MAC			1x / -	1x / -	2x / -	2x / 1x
	USB	FS DEV	FS DEV	FS OTG	FS OTG	FS OTG	FS OTG
	SD/MMC				✓	✓	✓
	USIC(UART, SPI, I <sup>2</sup> C, I <sup>2</sup> S)	4x	4x	4x	6x	6x	6x
	Ext. Memory I/F				✓	✓	✓
	CAN	2x	2x	2x	3x	6x	6x
	Touch Button	✓	✓	✓	✓	✓	✓

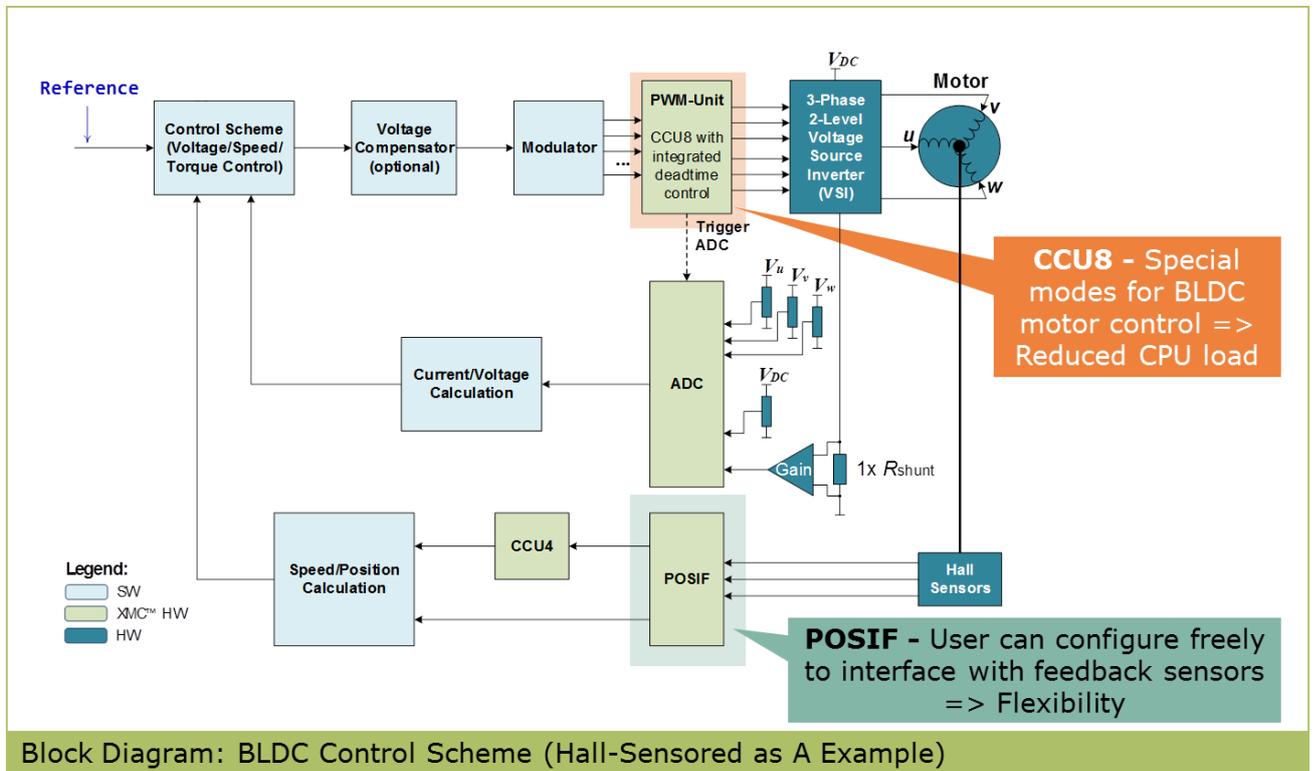
注：上面所写外设部分的通道数是最大值，其中各系列中又有很多 Part Number，每个 Part Number 的外设通道数或者存储器容量都不一样，具体请看芯片的数据手册。

### 3.3.3 产品应用领域

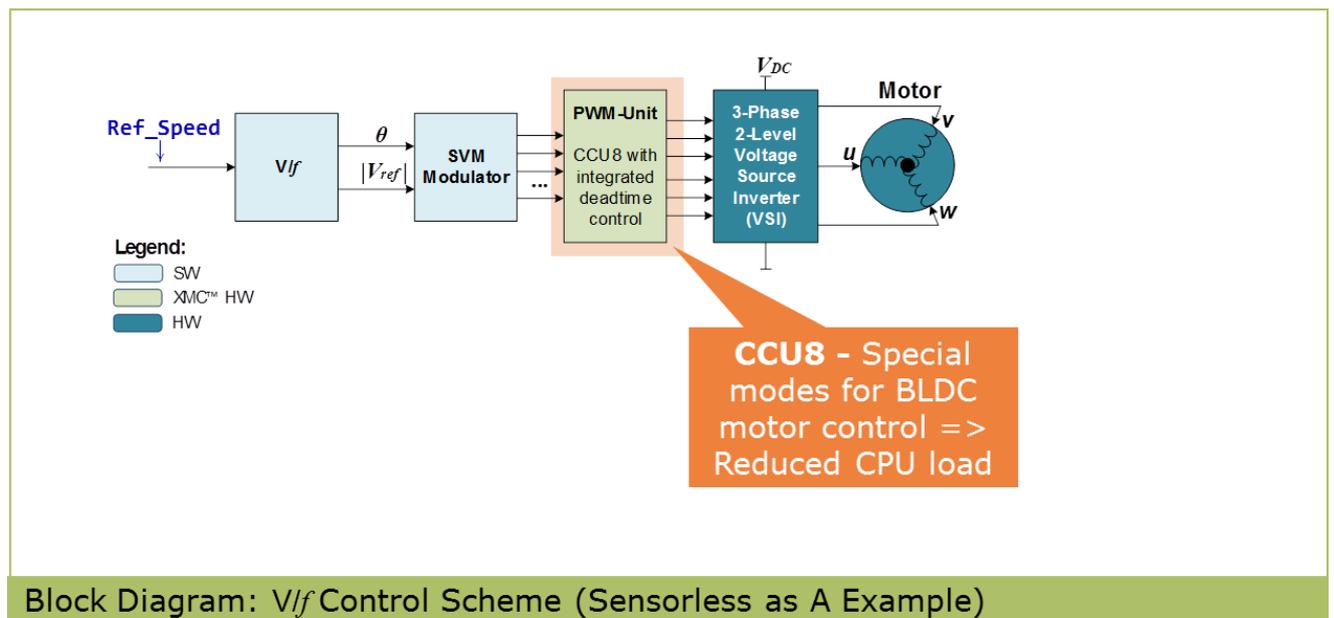
- 白色家电
  - 空调和冰箱的变频控制器
- 数字电源
  - PFC, LLC, 逆变器
- 通用变频器
- 通信模块
- 工厂自动化
- 工业机器人



下图是基于 XMC1000 的 3-Hall BLDC 方案模块图，CCU8 可以输出带死区的 3 对互补 PWM 波，PWM 可以触发 ADC，用来同步采样绕组电流。POSIF 则用来检测 3 相 Hall 信号。



下图是基于 XMC1000 的 V/f 方案模块图，CCU8 可以输出带死区的 3 对互补 PWM 波。



基于 XMC1000 的电机控制方案优势:

**FOC 方案:**

优势	具体细节
代码量少和运行时间短	代码量 < <b>16KB</b> , FOC 算法执行时间 < <b>20us</b>
减少 BOM 成本	ADC 内部集成 Gain, 可以省掉外部运放, 因此可以节省成本 <b>0.49-0.9€</b>
对参数依赖性减少	仅需 <b>1 个电感参数</b> 去测量 sensorless FOC 的转子角度和速度
直接无感 FOC 启动	需要的启动功率较小

**BLDC 方案:**

优势	具体细节
代码量少和运行时间短	代码量 < <b>12KB</b> , 算法执行时间 < <b>15us</b>
减少 BOM 成本	ADC 内部集成 Gain, 可以省掉外部运放, 因此可以节省成本 <b>0.49-0.9€</b>
系统更加强壮	POSIF 内部带 <b>Hall 信号的滤波器</b> , 无需外面的 RC 滤波电路

**V/f 方案:**

优势	具体细节
代码量少和运行时间短	代码量 < <b>8KB</b> , 算法执行时间 < <b>10us</b>
减少 BOM 成本	ADC 内部集成 Gain, 可以省掉外部运放, 因此可以节省成本 <b>0.49-0.9€</b>

**基于 XMC4000 的电机控制方案优势：**

**FOC 方案：**

优势	具体细节
集成 EtherCAT 的 XMC4800 和 XMC4300	无需外部 EtherCAT ASIC，外部存储器，从而减少成本
内部集成 Delta-Sigma 解调器	无需外部 Delta-Sigma 解调器
对参数依赖性减少	仅需 <b>1 个电感参数</b> 去测量 sensorless FOC 的转子角度和速度
直接无感 FOC 启动	需要的启动功率较小
超宽工作温度范围	支持高达 <b>125°C</b> 的工作温度

**BLDC 方案：**

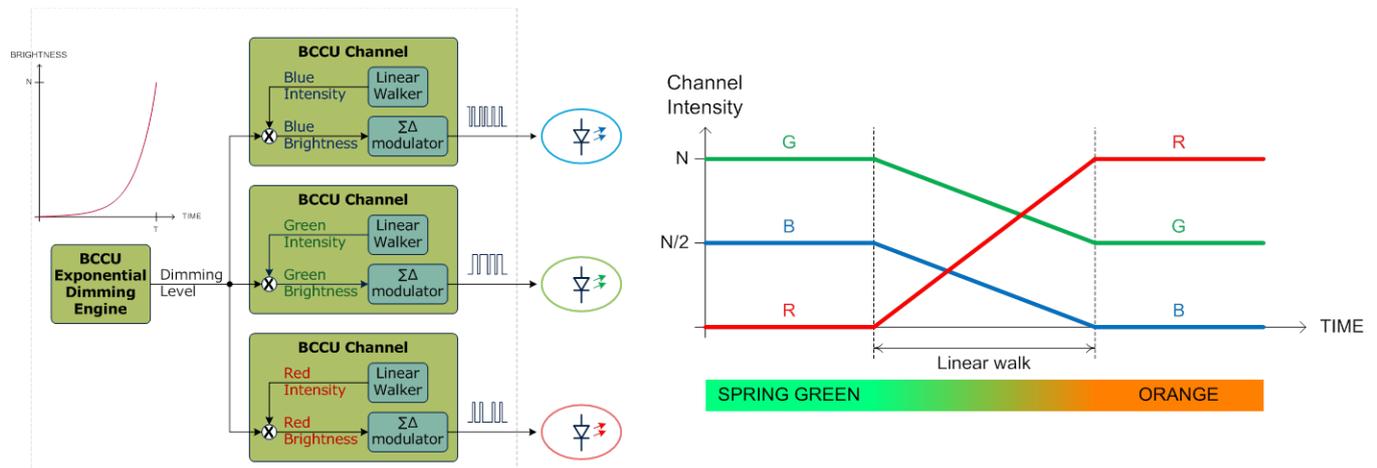
优势	具体细节
POSIF 减少 CPU 负担	可以检测 Hall 信号，增量式编码器信号， <b>减少 CPU 负担</b>
高速 ADC	多达 <b>4 个 ADC 内核</b> 可以同时采样，快速的 ADC 采样速度能使过零点电流能精确采样，从而使 PWM 的占空比能在准确时间更新， <b>减少的能量损耗</b> 。

**V/f 控制方案：**

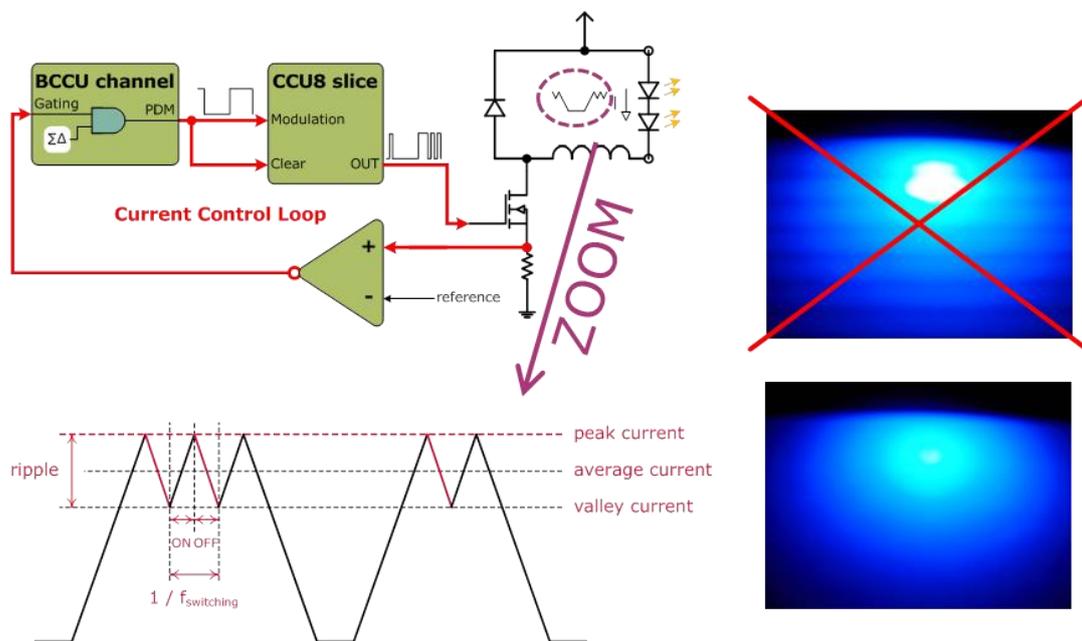
优势	具体细节
集成 EtherCAT 的 XMC4800 和 XMC4300	无需外部 EtherCAT ASIC，外部存储器，从而 <b>减少成本和 PCB 尺寸</b>
内部集成 Delta-Sigma 解调器	无需外部 Delta-Sigma 解调器， <b>减少 BOM 成本</b>

### 3.4.2 LED 控制方案

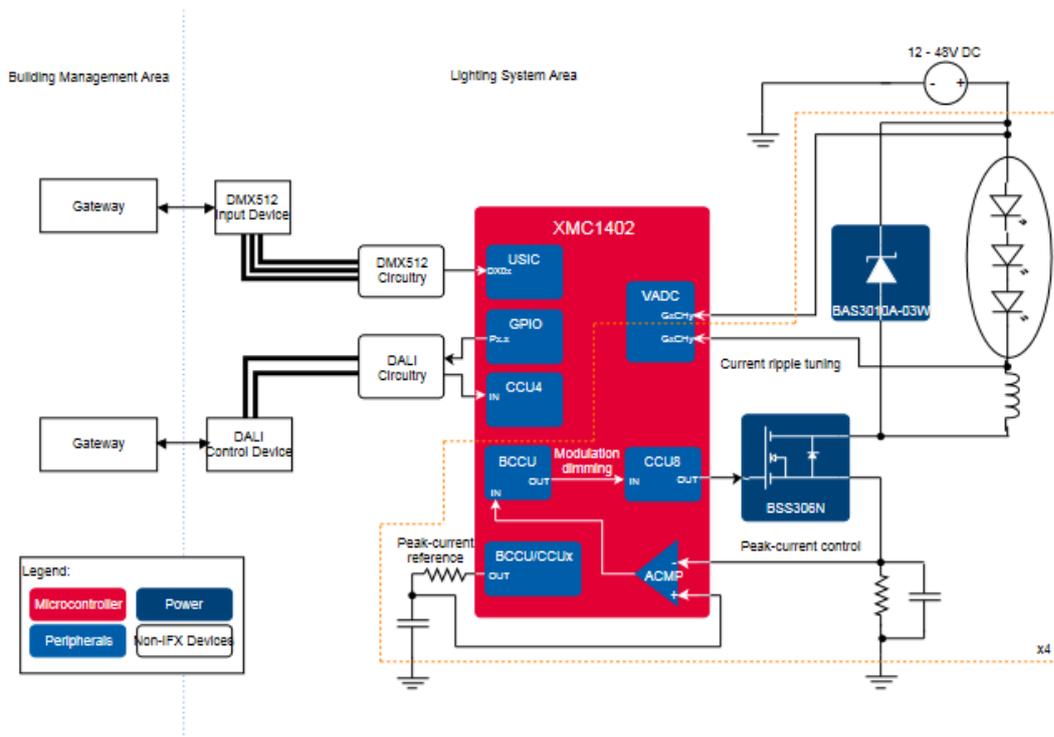
XMC1000 内部集成的 BCCU (Brightness Colour Control Unit) 模块可以实现硬件指数调光，内部的 Linear walker 机制能实现对色彩的顺滑调节。BCCU 单元有 9 个通道，每个通道可以控制 1 个 LED 或者 3 色 LED 的一种色彩，因此 BCCU 单元可以控制 3 个 RGB LED。



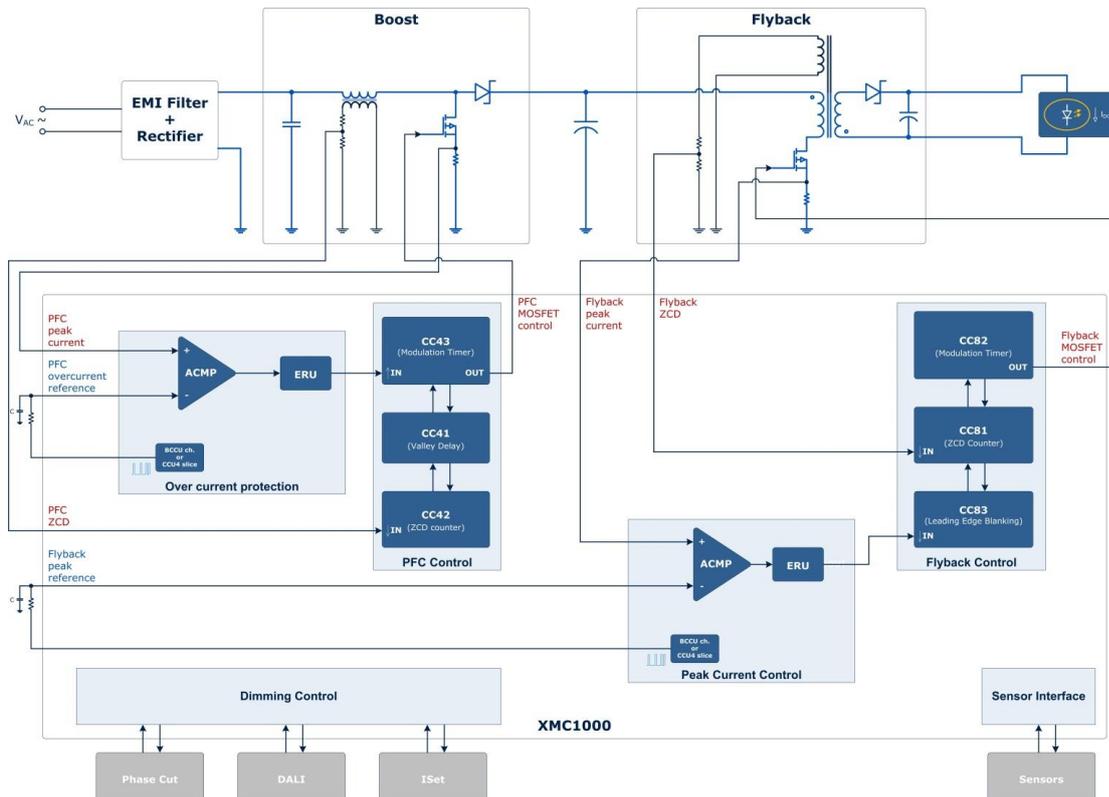
XMC1000 系列内部集成模拟比较器，可以实现快速的电流环控制。另外 BCCU 的输出结合 CCU8，可以实现 PDM (Pulse Density Modulation) 控制，使 LED 亮度调节变得很平滑和细腻。



英飞凌还提供 APP 和软件包支持 LED 照明应用中常用的通讯协议：DALI 和 DMX512.



在 LED 应用中往往也需要对输入的 AC 进行功率因素校正(PFC), XMC1000 系列 MCU 除了控制 LED 以外, 其它外设还可以实现 PFC 功能, 如下图所示。



**基于 XMC1000 的 LED 控制方案优势：**

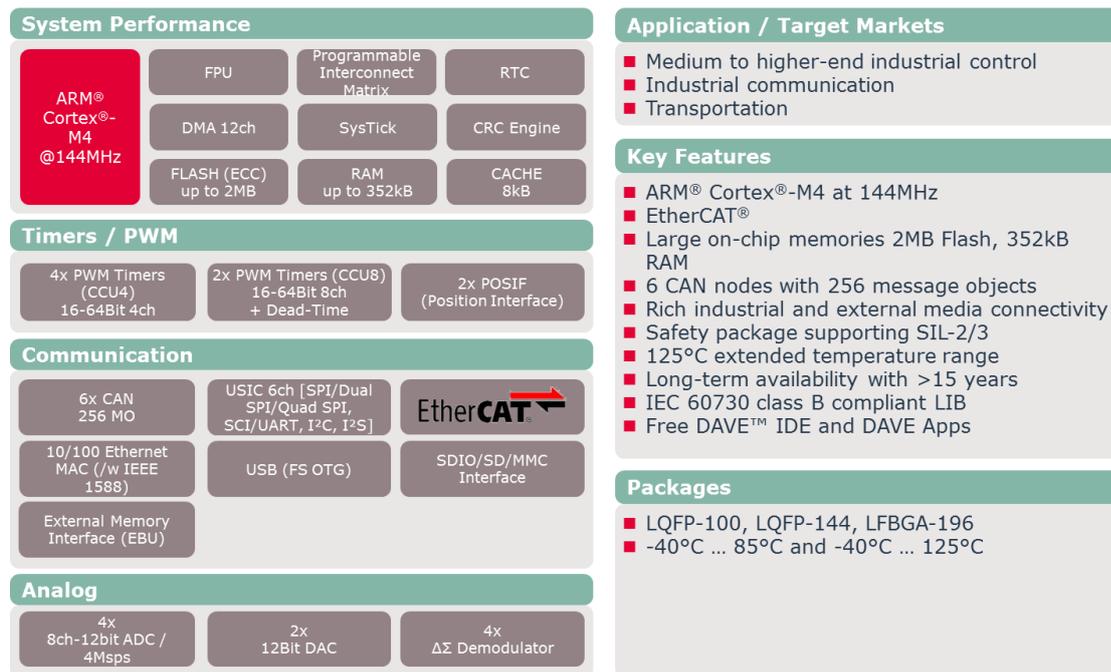
优势	具体细节
超低 CPU 负载	使用 BCCU 进行 LED 调光，模拟比较器进行电流比较
超高系统整合度	单一芯片实现：PFC，通讯，调光
通讯	支持 DALI, DMX512

### 3.4.3 EtherCAT 控制方案

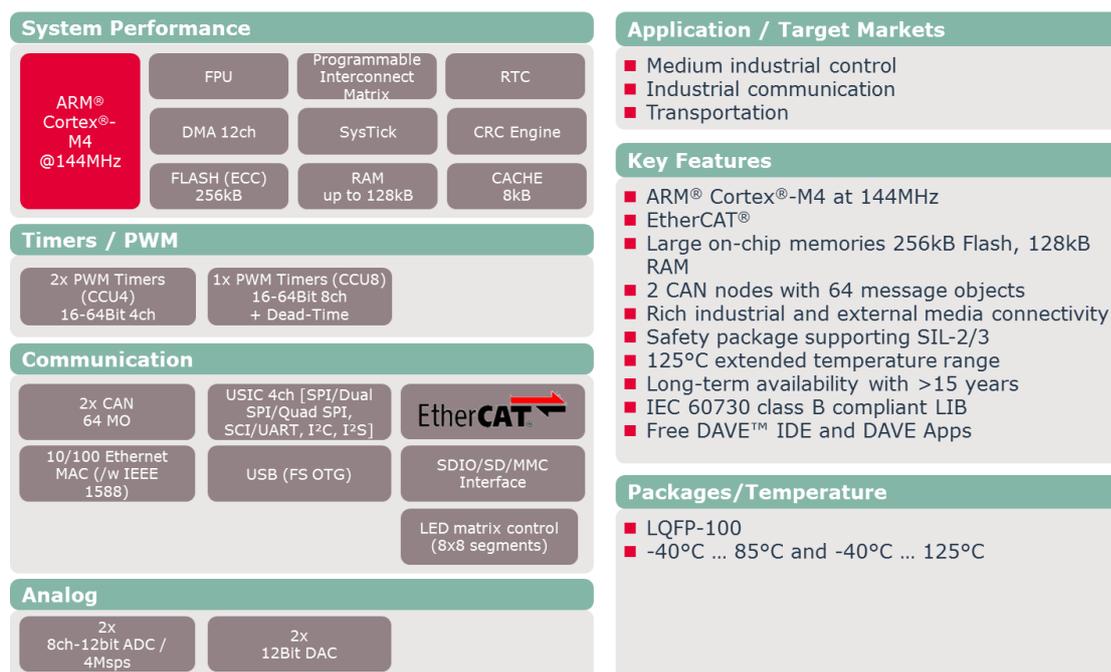
Ethernet for Control Automation (EtherCAT) 是一种基于以太网的高性能的工业控制总线。它的通讯速度可以达到 100Mbps, 在一个网络配置中最多支持 65535 个从站节点。它支持多种拓扑结构例如星型, 线型, 树形等。

英飞凌最新推出的 XMC4800 和 XMX4300 系列集成了 EtherCAT 模块。

#### XMC4800 资源模块图:



#### XMC4300 资源模块图:



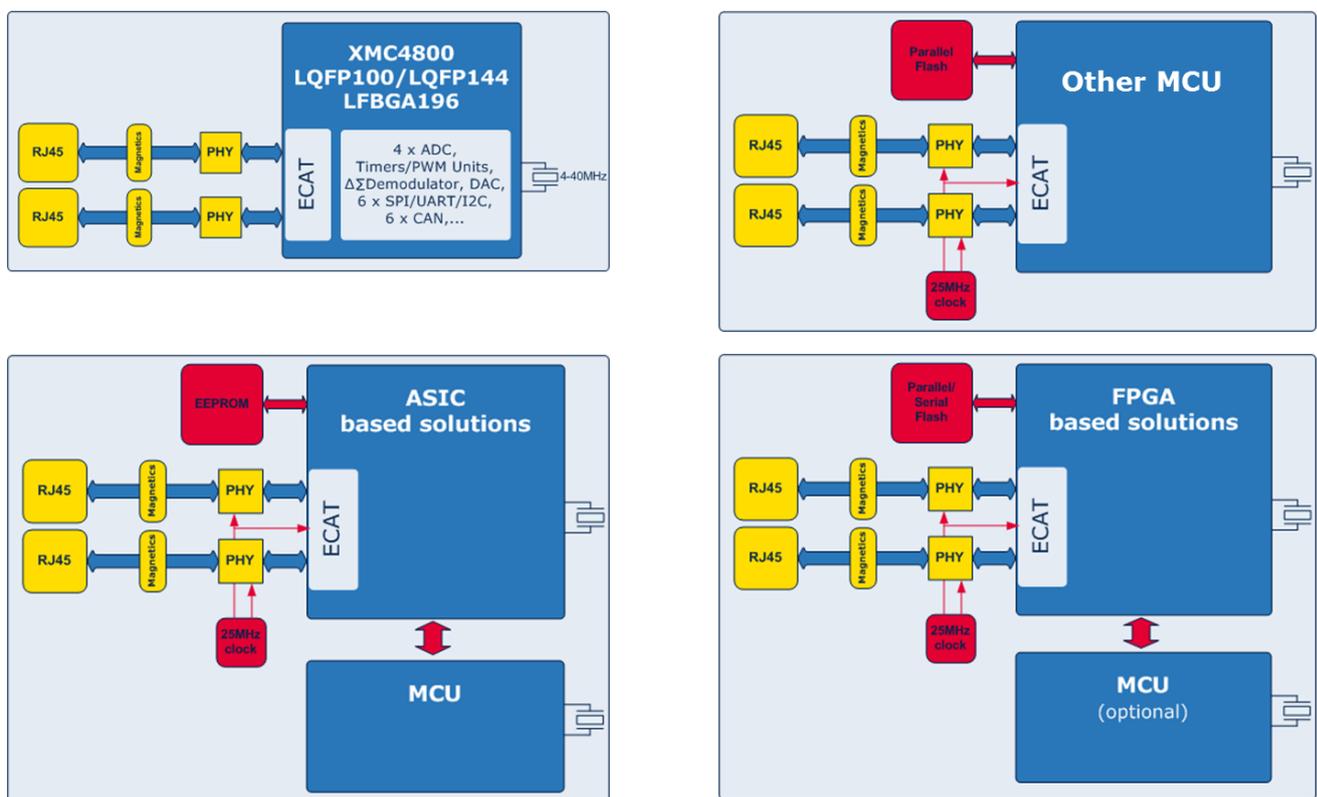
### 基于 XMC4800 的 EtherCAT 总线方案优势：

优势	具体细节
节省 BOM 成本	无需外部 EtherCAT 专用 IC, 无需外部 Flash 和 EEPROM, 无需外部晶振
减少 PCB 空间	单芯片解决方案能使终端产品体积更小
封装	LQFP 封装方便布线和 PCB 错误分析
集成电机控制单元	XMC4800 内置 2 单元 CCU8, 可以做 2 轴电机控制
兼容性	100%兼容典型 EtherCAT 专用 IC ET1100

XMC4800 内部集成了电机控制模块和模拟信号检测模块，可以用于电机控制领域。XMC4300 刨去了 XMC4800 中电机应用模块，可用于工业控制中的 I/O 模块。

以下是基于 XMC4800 的 EtherCAT 控制方案和其它方案比较。

左上是基于 XMC4800 的 EtherCAT 控制方案，可以看到这个方案仅需 2 片外部 PHY，无需外部 EEPROM 和外部存储器。右上是基于其它带 EtherCAT 模块的 MCU，大部分这类的 MCU 都需要外部的 Flash 和 EEPROM，而且 PHY 需要外部的晶振。左下是基于 ASIC（例如 ET1100）的解决方案，这种方案需要用额外的 MCU 去做控制，而且同样需要外置的 EEPROM 和外部晶振。右下是基于 FPGA 的解决方案，这种方案往往采样高性能的 FPGA 实现 EtherCAT IP，除此之外，有些 FPGA 还可以做主控，这种方案虽然运算性能和扩展性都较好，但是成本很高，而且同样需要外部的存储器和外部晶振。



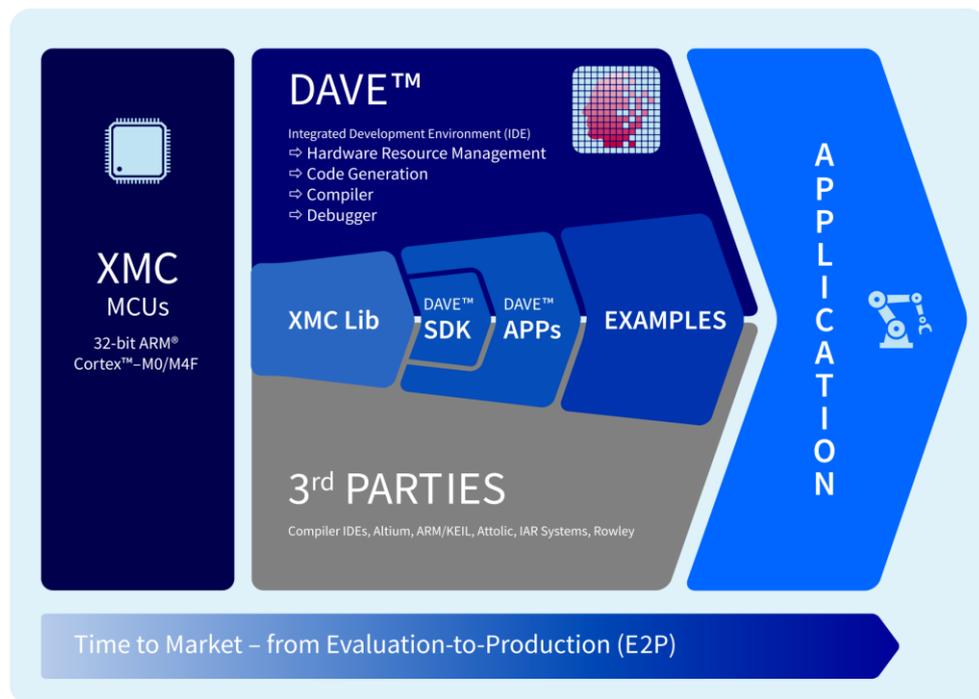
## 4 集成开发环境 (IDE)

### 4.1 DAVE4

DAVE4 是英飞凌公司开发的基于 Eclipse 的免费开发环境。除了一般的代码编辑和调试功能外，它内部还集成了很多 APP，这些 APP 都是基于控件的形式提供给用户的，用户仅需在自己的应用添加需要的 APP 控件，再双击控件就可以在该控件的 GUI 中设置相关参数，然后自动生成代码，接着就可以使用该 APP 的功能了。目前发布了 DAVE 版本中基本涵盖了常用外设的 APP。

另外，DAVE4 中包含了 XMC Lib (Low Level Driver)，这个基层驱动库是经过完全测试且兼容 CMSIS/MISRA 2004 标准，支持所有 XMC 产品。在 DAVE4 中新建的工程都包含了 XMC Lib。

在 DAVE4 下开发的工程也可方便的移到到第三方的开发环境（例如 Keil，IAR），下面章节用详细介绍移植流程。



#### 4.4.1 安装

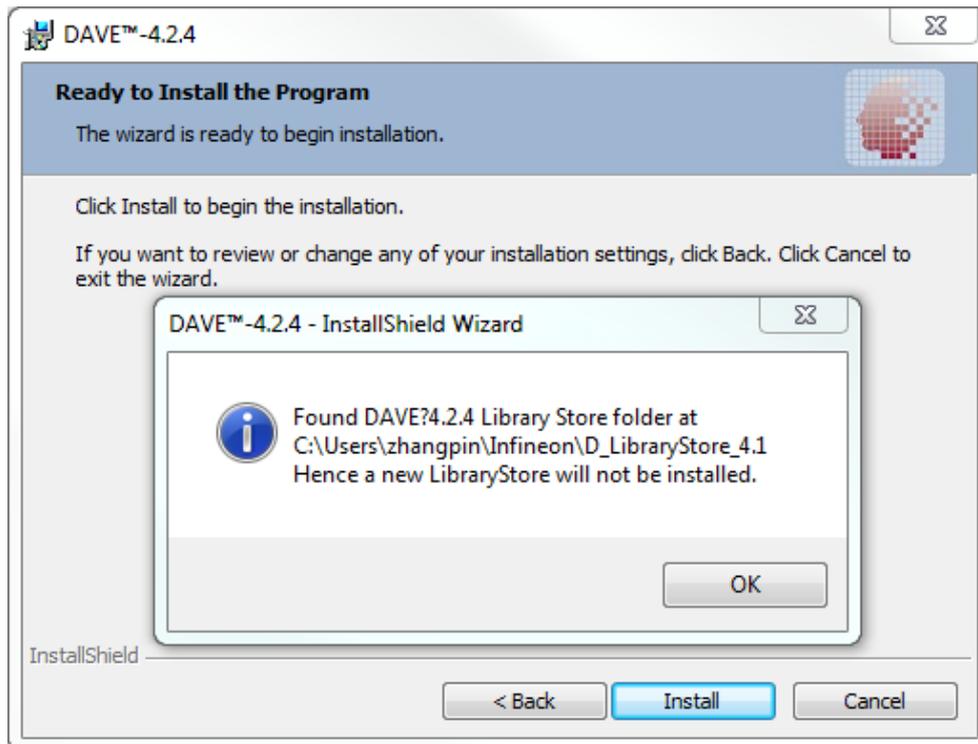
**DAVE4 下载地址:**

<http://www.infineon.com/dave>

### DAVE4 安装准备:

对于以前未安装 DAVE4 版本的用户，可直接进行安装。对于之前安装过 DAVE4 的用户需要删除库所在的文件夹 (C:\Users\yourname\Infineon\D\_LibraryStore\_4.1)。

如果没有删除库文件夹，直接安装，则可能会出现下面问题。导致这个问题的原因是安装文件检测到了 D\_LibraryStore\_4.1 的存在，因此没有安装新的库文件。这个会导致新建工程时有些新的 MCU 型号找不到。



### DAVE4 安装问题:

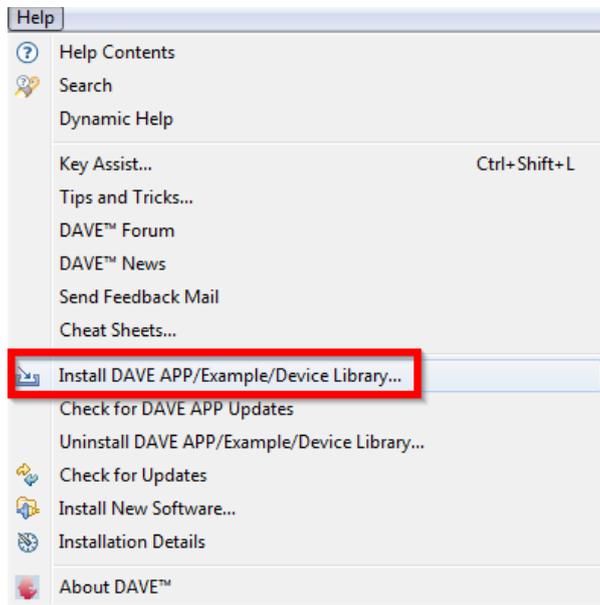
**问题一：在 DAVE4 安装完后，可以会出现下面问题：**

- 新建工程没有反应
- 新建工程无法选择相应的器件
- 新建工程无法选择相应的 APP

**解决办法：**

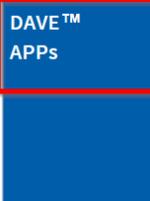
出现这些问题的原因大部分都是因为 DAVE 的库文件没有安装完全。在这种情况下可以安装更新库，具体步骤如下：

1) 选择 Help -> Install DAVE APP/Example/Device Library



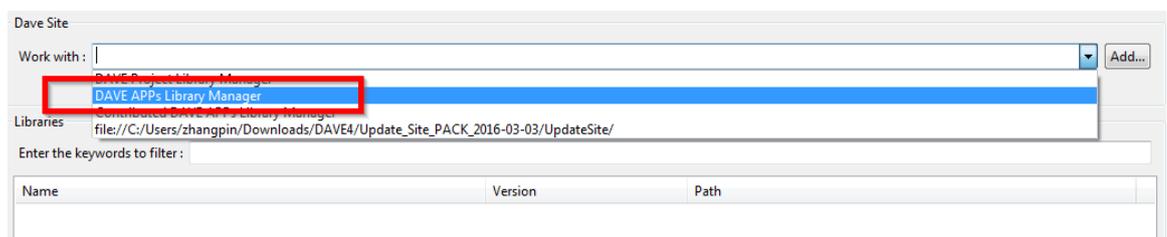
2) 在弹出窗口中，下拉 Work with，这里提供了在线安装和离线安装两种方法。如果是要离线安装，则需要首先下载离线安装包：

<http://www.infineon.com/dave>

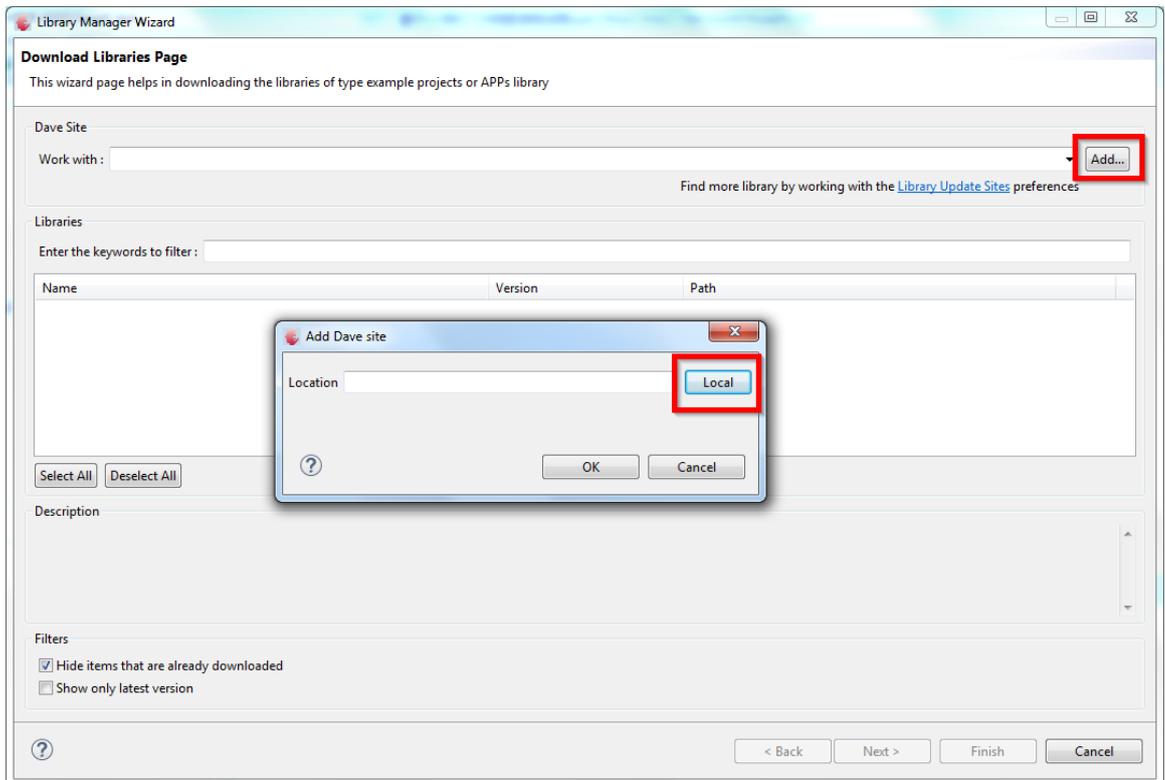
	 Download	Graphical User Interface (GUI) configurable application oriented software components using Level Driver); arranged in a library (APIs) <a href="#">DAVE™ Release Note APPS</a>			
		<table border="1"><thead><tr><th>General Purpose and System DAVE™ APPS:</th><th>Application Specific DAVE™ APPS:</th></tr></thead><tbody><tr><td> </td><td> </td></tr></tbody></table>	General Purpose and System DAVE™ APPS:	Application Specific DAVE™ APPS:	
General Purpose and System DAVE™ APPS:	Application Specific DAVE™ APPS:				

下载完后，将其解压到本地磁盘。**注意：路径中不能有中文或者空格。**

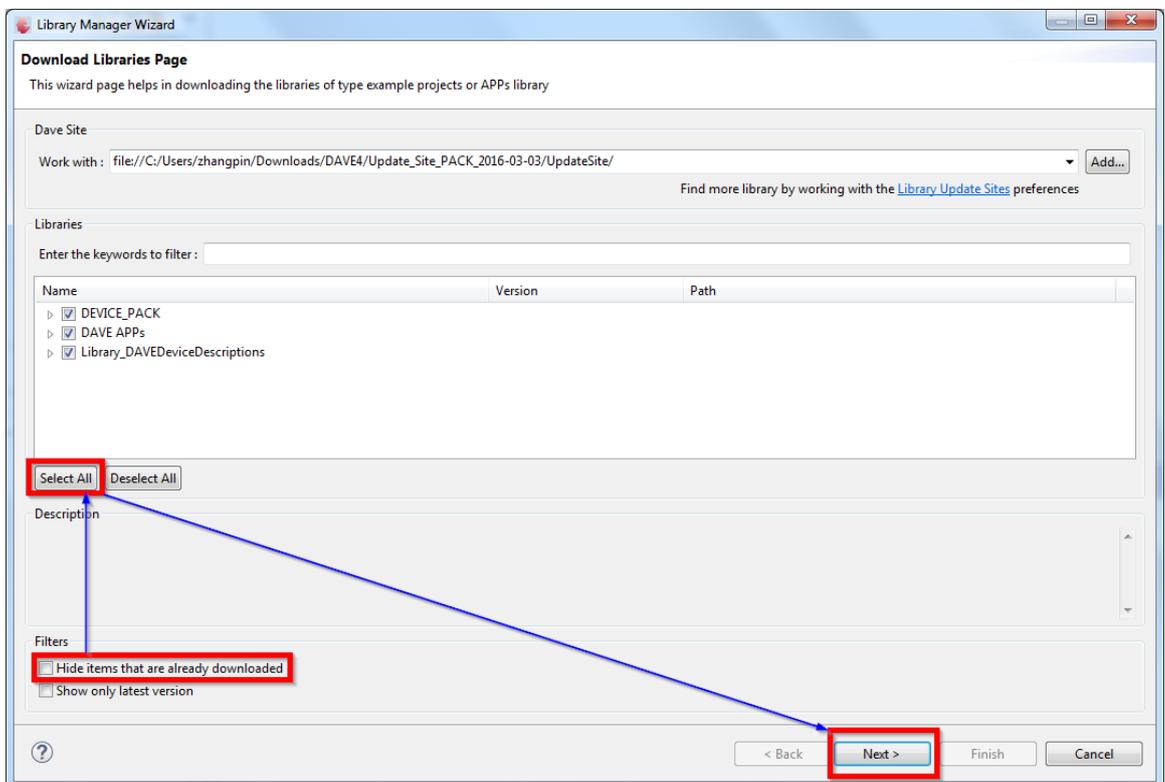
3) 如果选择在线安装，则选择 DAVE APPs Library Manager



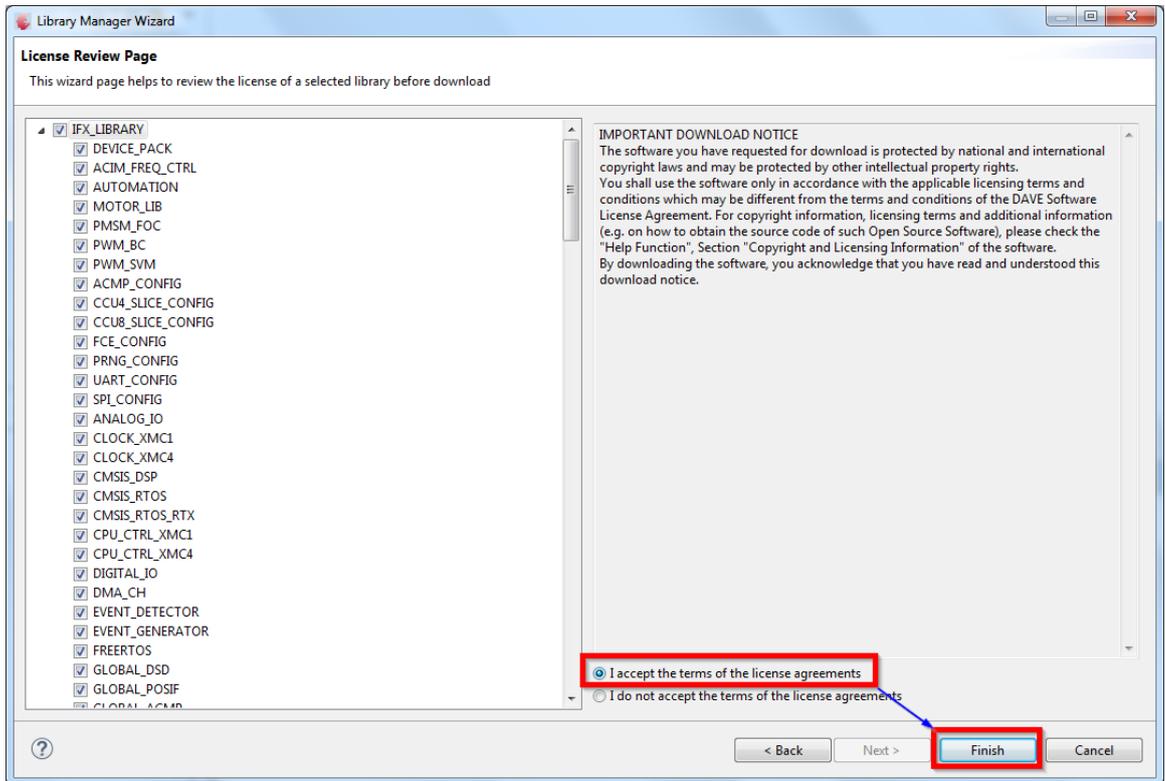
4) 如果选择离线安装，点击 **Add, Local** 将离线安装包的解压目录添加进来



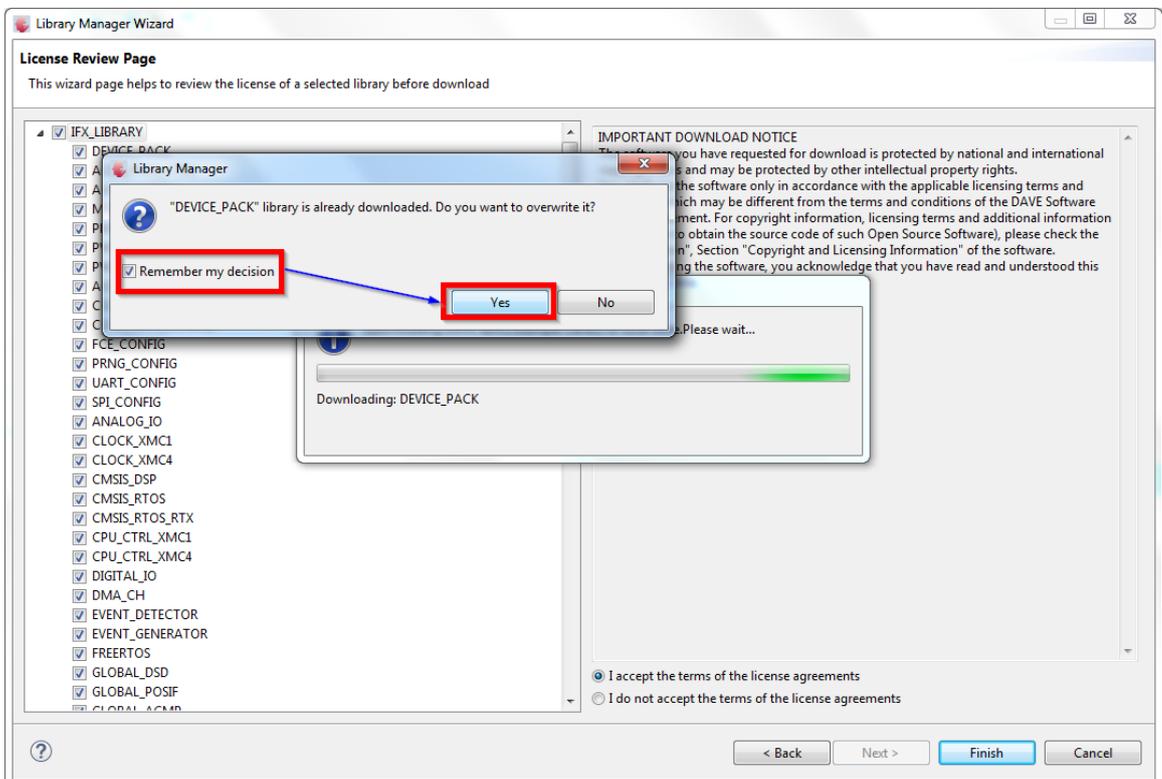
5) 在线/离线安装



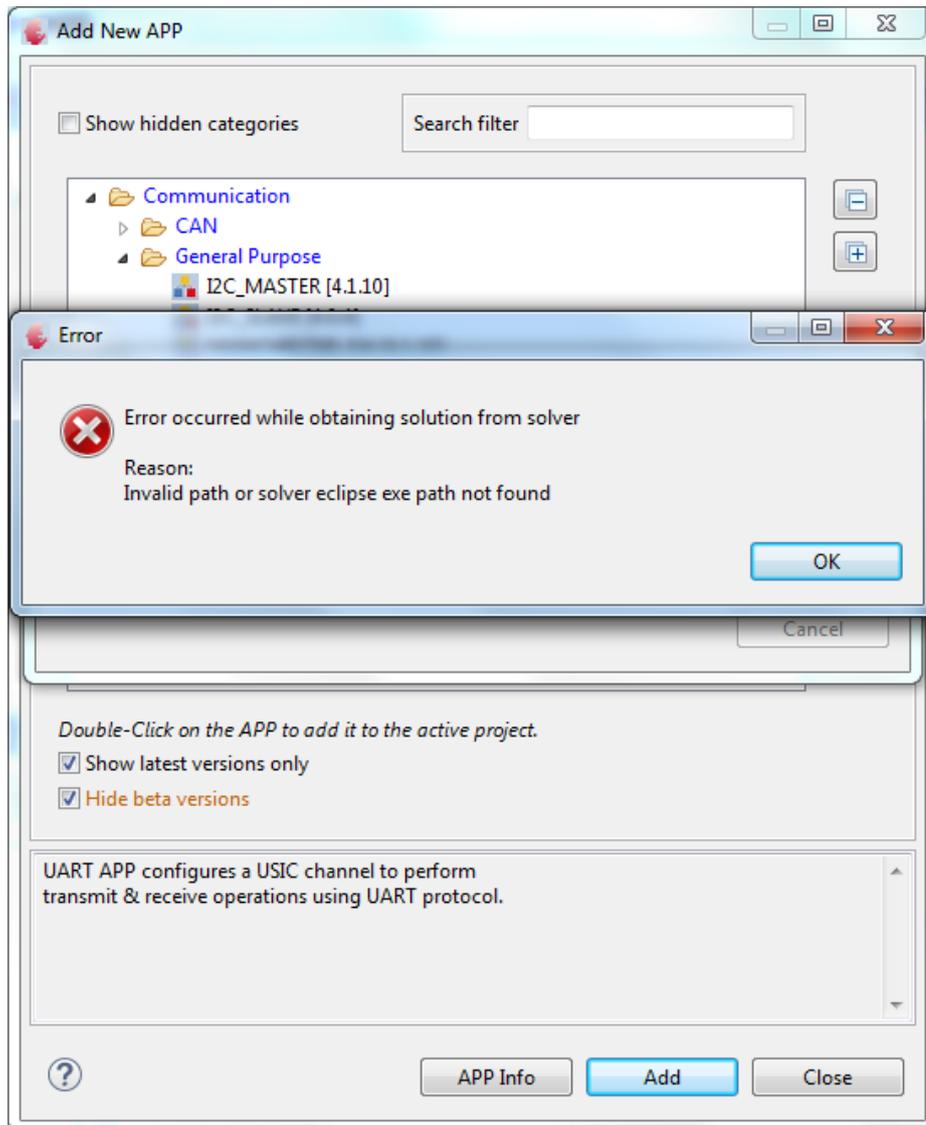
6) 全选，覆盖



7) 如有提示，选择全部覆盖安装



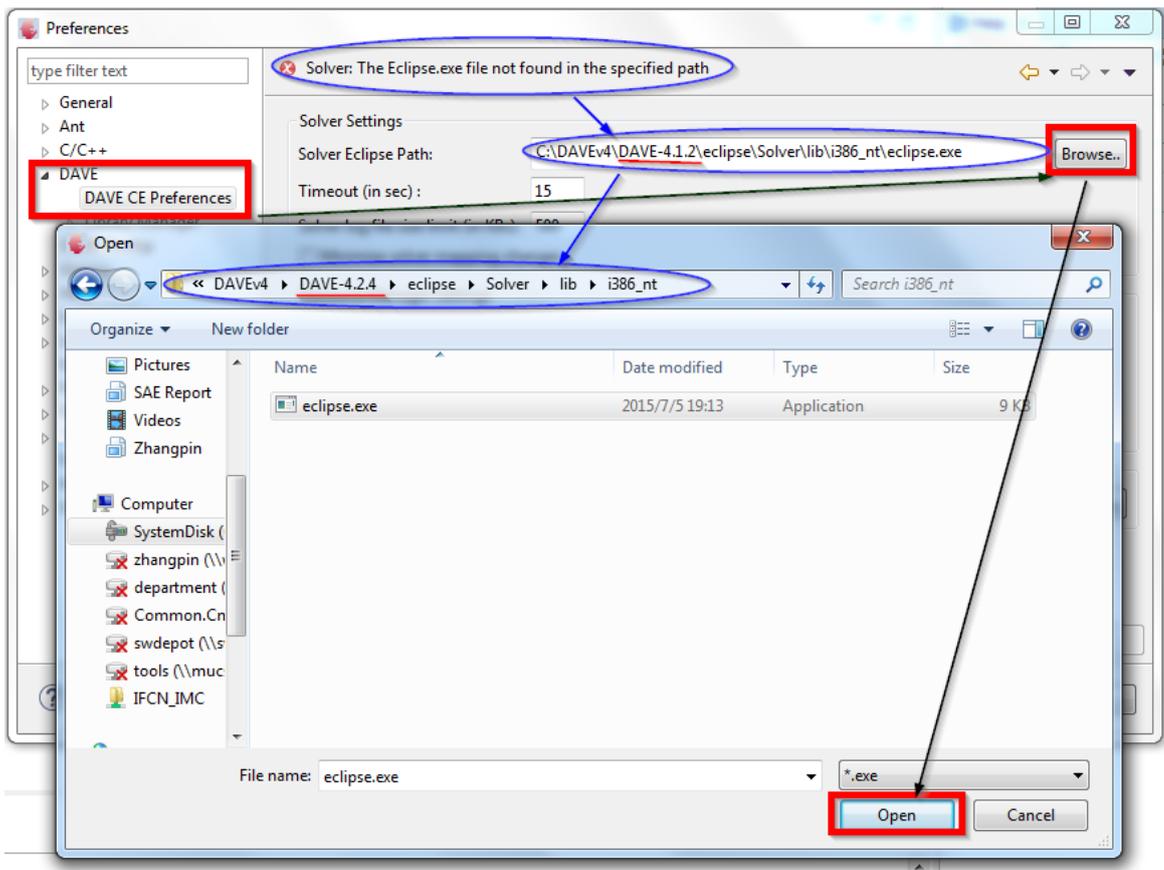
问题二：在添加 APP 时，也可以出现找不到 Solver 路径的问题。



**解决办法：**

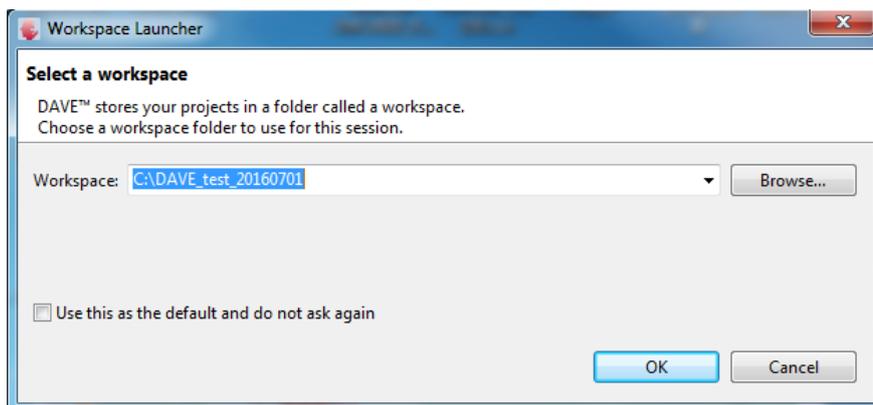
这个问题的原因是 APP 的路径设置有可能有误。重新设置路劲：

- 1) Window->Preferences
- 2) Dave ->Dave Preference, 找到正确的安装路径，并设置

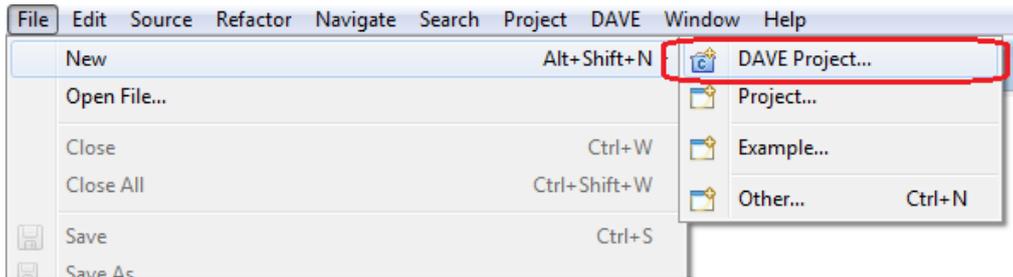


## 4.4.2 新建工程

### 1) 新建 workspace

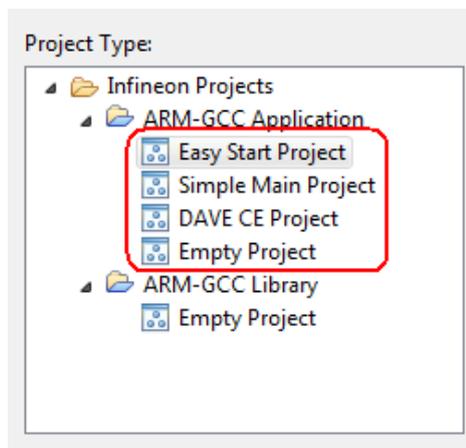


## 2) File -> DAVE Project



## 3) 可以根据需要，选择创建下面 4 种工程中的一种。

- **Easy Start Project:** 可以运行在英飞凌开发套件上实现简单功能的一个例程
- **Simple Main Project:** 包含底层驱动库的一个空工程 (含 main.c)
- **DAVE CE Project:** 与 Simple Main Project 类似，但可以添加 DAVE APP
- **Empty Project:** 真正的空工程（不含任何文件），仅设置工程中的预定义路径

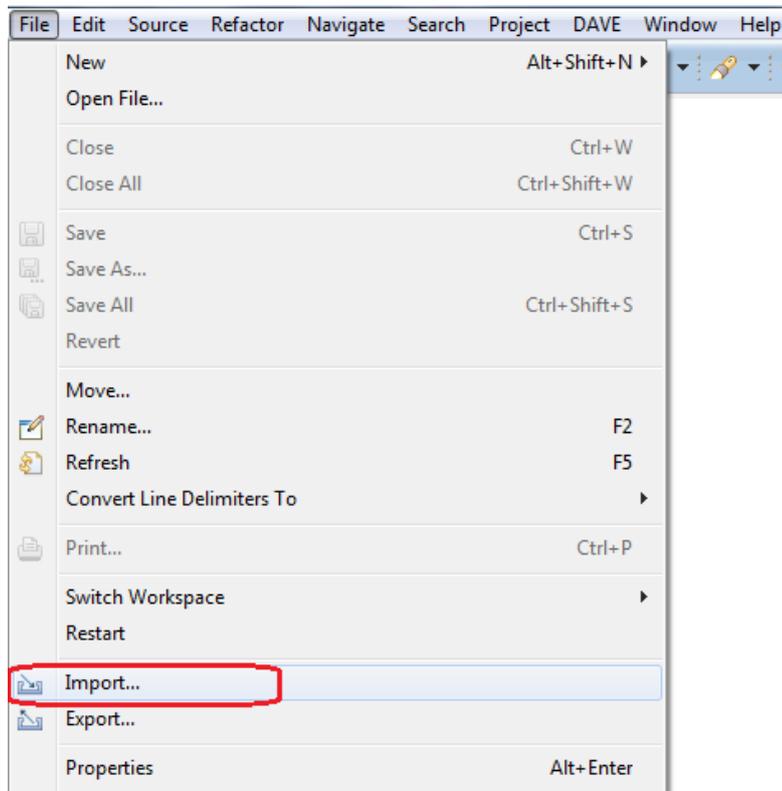


## 4) 选择芯片型号，生成工程。

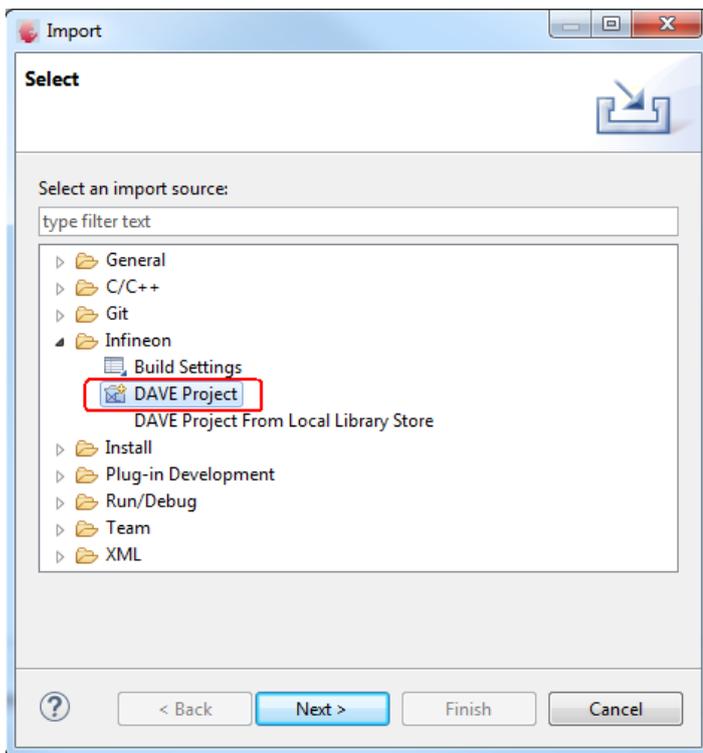
### 4.4.3 导入工程

在 DAVE4 中不能直接打开现有的工程，需要首先打开 DAVE workspace，然后导入现有的工程文件。

1) File -> Import



2) 选择 Infineon -> DAVE Project

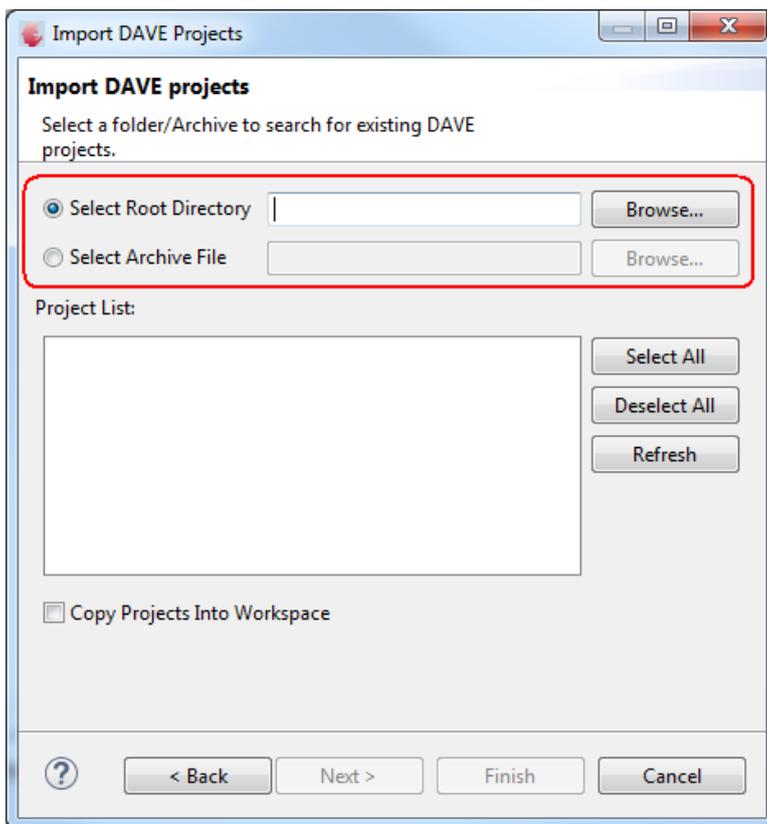


3) 这里提供了两种方法导入工程.

- **Select Root Directory:** 选择工程文件夹的路劲
- **Select Archive File:** 选择工程压缩文件的路径

选择 **Select Root Directory**, 则直接打开工程文件夹, 接着在 **DAVE** 中对文件进行编辑将直接修改原工程文件。

选择 **Select Archive File**, 则 **DAVE** 首先将解压工程, 然后将解压后的工程导入当前 **workspace**。这个方法支持一次导入多个工程。(即可以将多个工程压缩打包成一个文件, 然后用这个方法导入工程)

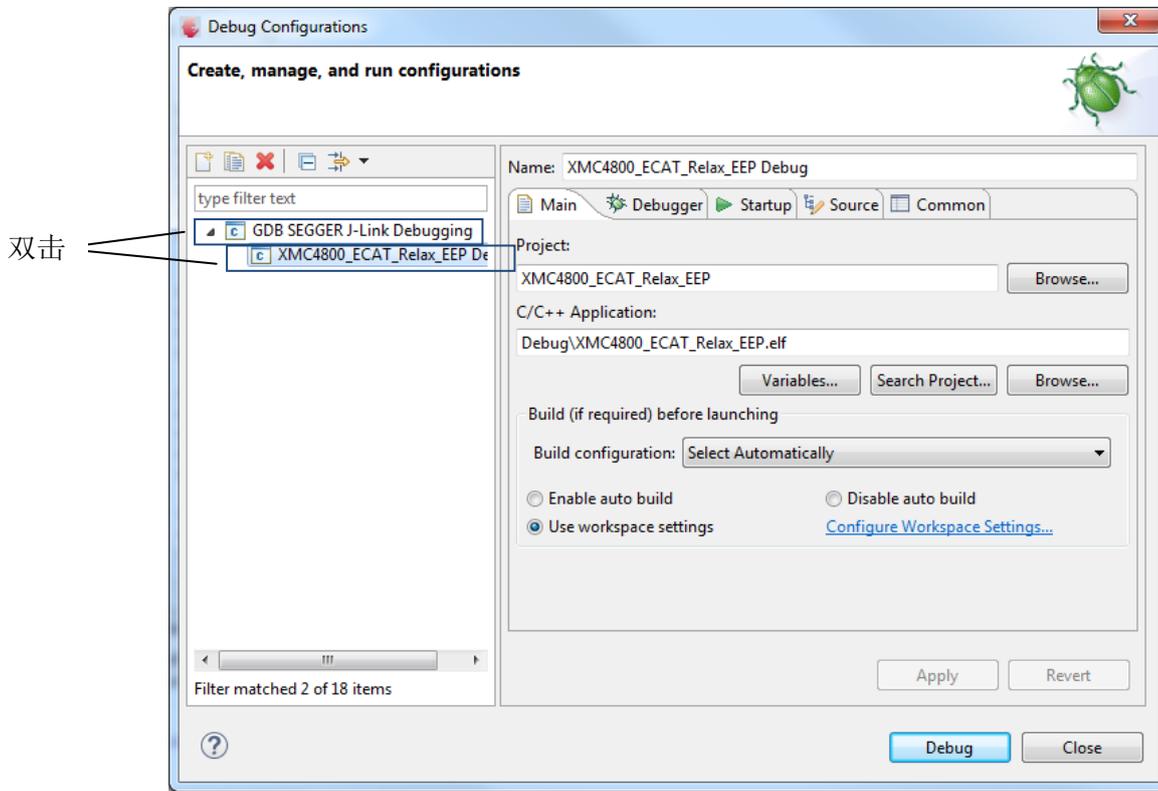


#### 4.4.4 调试技巧

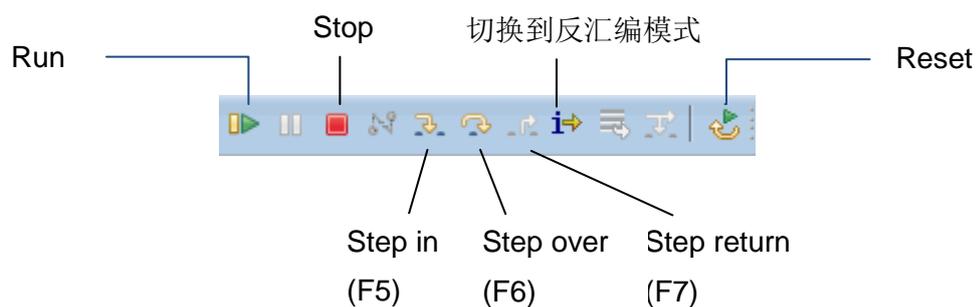
##### 编译和开始调试:



编译完后，点击，开始调试。当第一次调试时，会弹出下面对话框。接着双击 **GDB SEGGER J-Link Debugging**，新建了一个调试配置，接着再双击新建的配置就可以开始调试了。



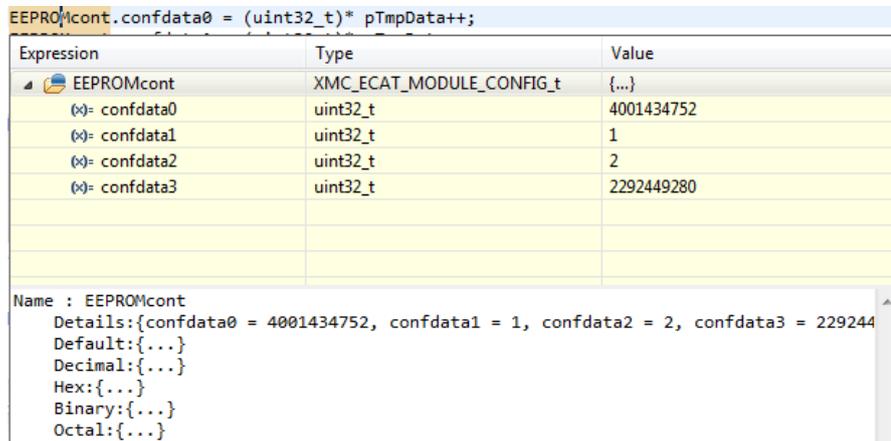
### 调试工具条：



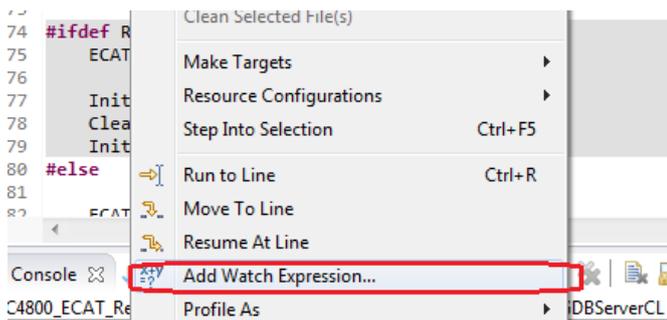
### 查看变量：

有几种方法可以查看变量。

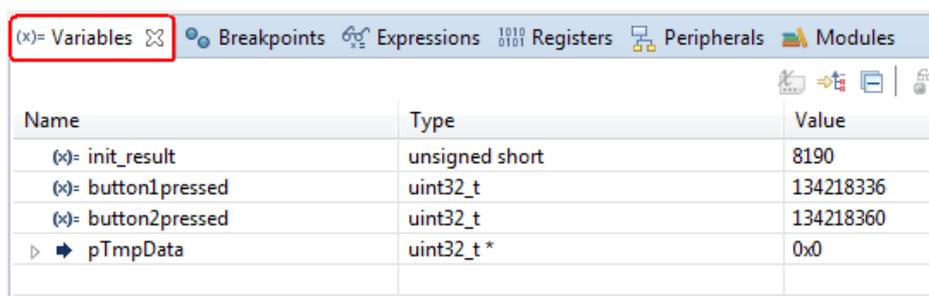
1) 可以把光标直接放在变量名上



2) 双击选中变量名，右击，选择“Add Watch Expression”



3) 如果是局部变量，则直接可以在右边窗口中“Variables”栏中查看



### 查看外设寄存器:

先选择右边窗口中 Peripheral 栏，勾选需要查看的外设

(x)= Variables   Breakpoints   Expressions   Registers   **Peripherals**   Modules

Peripheral	Address	Description
DSD_CH0	0x40008100	Delta Sigma Demodulator
DSD_CH1	0x40008200	Delta Sigma Demodulator
DSD_CH2	0x40008300	Delta Sigma Demodulator
DSD_CH3	0x40008400	Delta Sigma Demodulator
EBU	0x58008000	External Bus Unit
<input checked="" type="checkbox"/> ECAT0	0x54010000	EtherCAT 0
ECAT0_CON	0x500041B0	EtherCAT 0 Control Register
ECAT0_FMMU0	0x54010600	EtherCAT 0
ECAT0_FMMU1	0x54010610	EtherCAT 0

打开外设寄存器查看窗口

ECAT0: 0x54010000   New Renderings...   ECAT0

Register	Address	Value
ECAT0	0x54010000	
AL_CONTROL	0x54010120	0x0000
AL_EVENT_MASK	0x54010204	0x00000000
AL_EVENT_REQ	0x54010220	0x00000000
AL_STATUS	0x54010130	0x0000
AL_STATUS_CODE	0x54010134	0x0000
BUILD	0x54010002	0x0000
DC_ACT	0x54010981	0x00
DC_ACT_STAT	0x54010984	0x00

修改代码后重新载入:

有时想在调试的时候修改代码，改完代码后可以右击左上“main() at...”，选择“Terminate and Relaunch”。

Thread #1 <main> (Suspended; Breakpoint)

**main() at main.c:45 0x80078**

- Copy Stack   Ctrl+C
- Find...   Ctrl+F
- Drop To Frame
- Step Into   F5
- Step Over   F6
- Step Return   F7
- Instruction Stepping Mode
- Use Step Filters
- Resume Without Signal
- Resume   F8
- Suspend
- Terminate   Ctrl+F2
- Terminate and Relaunch**

```

main.c   linker_script.ld
int main(void)
{
    UINT16 init_result; // I
    uint32_t button1pressed;

```

### 退出调试:

点击，退出调试。一定要先退出调试，然后才能再次开始调试，否者将会报错。

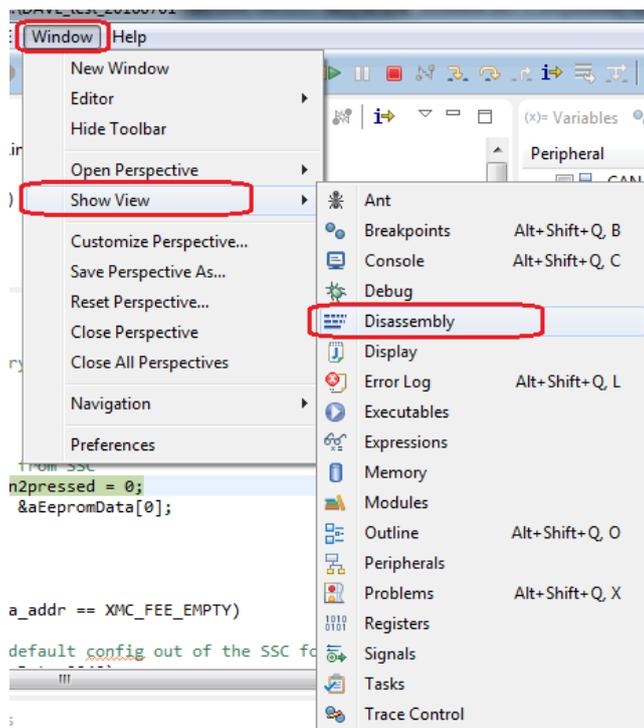
### 界面切换:

编辑界面和调试界面可以相互切换。



### 查看和运行反汇编程序:

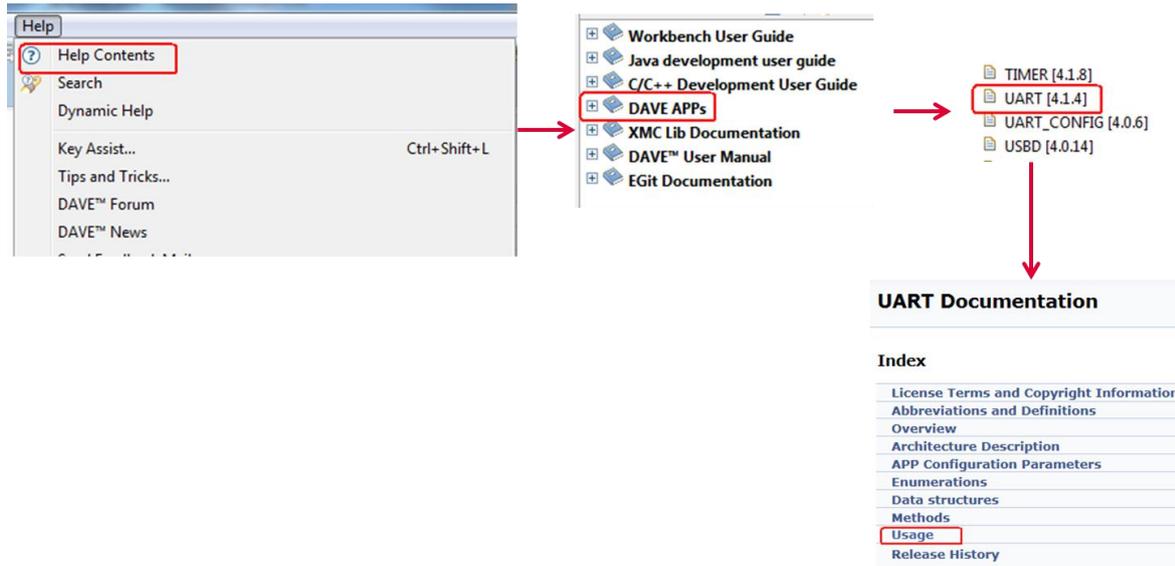
打开 Window->Show View->Disassembly



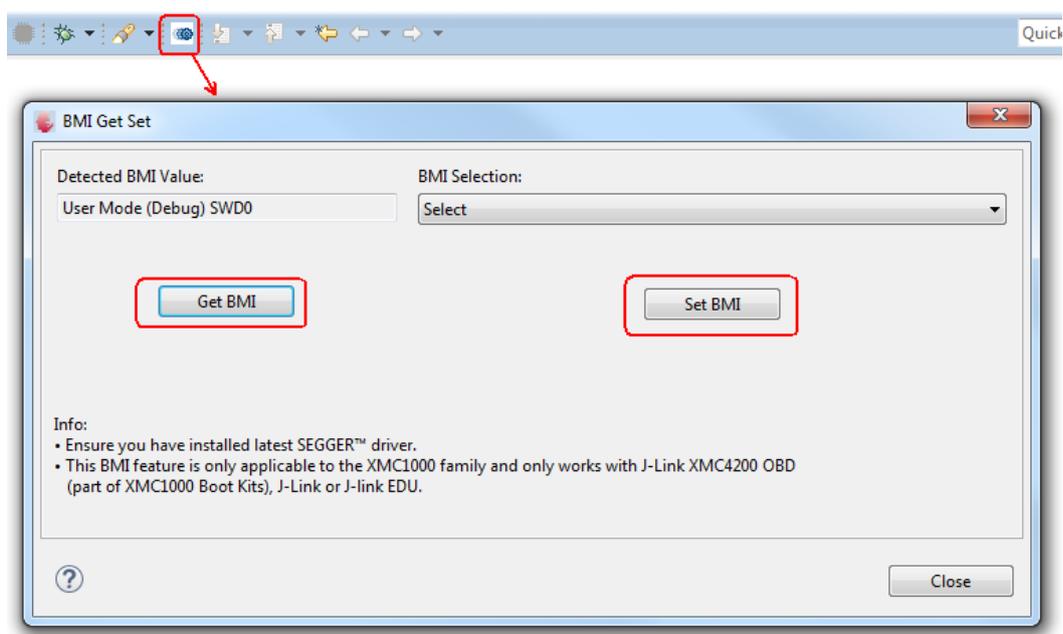
点击, 然后使用 F5, 就可以开始单步调试反汇编代码了。

#### 4.4.5 使用 APP 流程

在帮助文档中可以找到 APP 的介绍和使用说明。以 UART APP 为例，可以打开 APP 的帮助文档来查看如何使用这个 APP，如下图所示。



#### 4.4.6 读取和修改 BMI

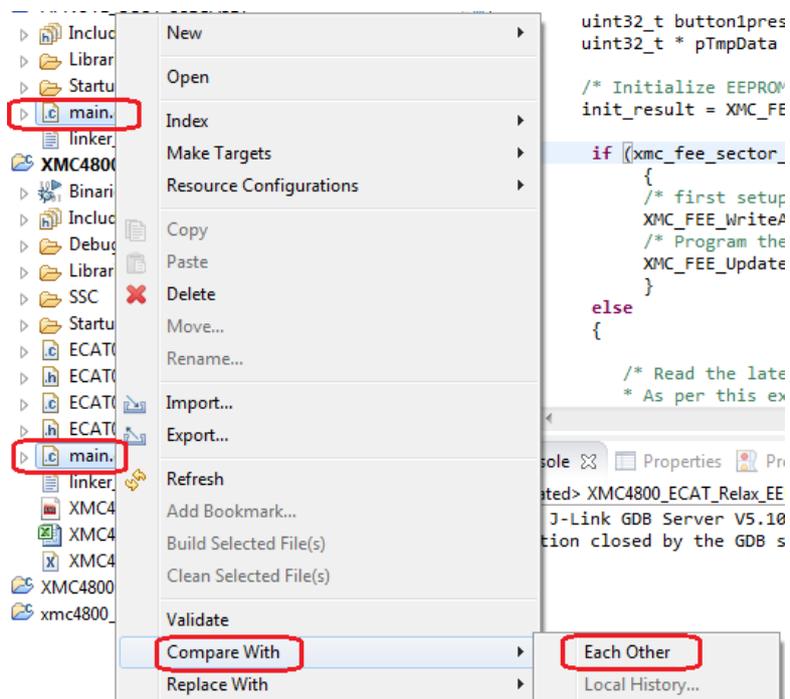


#### 4.4.7 快捷键

功能说明	快捷键	界面
查看结构体或函数原型	F3	编辑界面
多行注释	Ctrl + /	编辑界面
在工程中查找	Ctrl + H	编辑界面
Step in	F5	调试界面
Step over	F6	调试界面

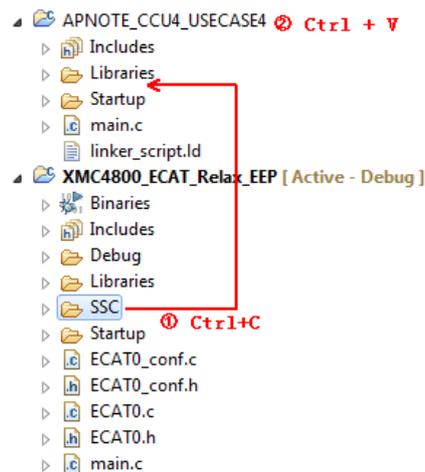
#### 4.4.8 文件比较功能

DAVE 还支持文件比较功能，首先按住 **Ctrl**，选中不同工程中两个同名的文件，右击选择 **Compare With -> Each Other**



#### 4.4.9 文件拷贝功能

DAVE 中支持两个工程中的文件或文件夹相互拷贝功能，如下图所示，可以选中一个工程中的一个文件夹，然后 **Ctrl+C**，再选中目标工程，**Ctrl+V**，这样这个文件夹就可以拷贝到目标工程中去。



#### 4.4.10 代码和变量定位

1) 如何把代码放在 RAM 区域?

■ 如果需要把某个函数放在 RAM 中运行，可以使用 `__attribute__` 关键字，例如：

```
void __attribute__((section (".ram_code"))) CalSumInRAM(void)
{
    ...
}
```

`ram_code` 这个段是 RAM 中用于放置 RAM code 的一个段，预先在 `ld` 文件已经定义好了，不需要用户自己定义。

- 如果需要把一个文件放在 RAM 中运行，则需要修改 ld 文件，首先把这个文件从 text 段中 exclude 出来，然后再添加到 ram\_code 段中。例如把 test1.c 这个文件放在 RAM 中运行，可以如下操作：

```
.text :
{
    ...
    *(EXCLUDE_FILE( *test1.o) .text.* .gnu.linkonce.t.*);
    ...
} > FLASH

.ram_code : AT(DataLoadAddr + __data_size)
{
    __ram_code_start = .;
    /* functions with __attribute__ ((section (".ram_code"))*/
    *(.ram_code)

    /* list the object files running from PSRAM here*/
    *test1.o(*text.*);

    . = ALIGN(4);
    __ram_code_end = .;
} > SRAM
```

### 2) 如何把变量放在 RAM 中指定位置？

在 ld 文件中定义一块自己的数据区，然后在源文件中就可以用 \_\_attribute\_\_ 关键字把变量放在这个区域的首地址了。

例如把一个变量放在 0x20002000 这个地址：

在 ld 文件定义一块从 0x20002000 开始的数据区：

```
.myDataBlock 0x20002000:
{
    *(.myDataSection)
}> SRAM
```

在源文件中把变量 sum 放在 0x20002000 的地址：

```
uint8_t __attribute__ ((section (".myDataSection"))) sum;
```

### 3) 如何把 const data 放在 Flash 中的指定位置？

在 ld 文件中定义一块自己的常量区，然后在源文件中就可以用 \_\_attribute\_\_ 关键字把变量放在这个区域的首地址了。

例如把一些常量放在 0x10011000 开始的地址：

在 `ld` 文件定义一块从 `0x10011000` 开始的常量区：

```
.myConstBlock 0x10011000:
{
    KEEP(*(.myConstSection)) /* keep my const even if not referenced */
}> FLASH
```

在源文件中把常量 `myconst` 数组放在 `0x10011000` 的地址。

```
const uint8_t __attribute__((section(".myConstSection")))
myconst[8] = {'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h'};
```

#### 4) 如何把代码放在 `Flash` 中的指定位置？

在 `ld` 文件中定义一块自己的代码段，然后在源文件中就可以用 `__attribute__` 关键字把代码放在这个区域的首地址。

例如把一个函数放在 `0x10016000` 开始的地址：

在 `ld` 文件定义一块从 `0x10016000` 开始的代码区：

```
.myCodeBlock 0x10016000:
{
    KEEP(*(.myCodeSection)) /* keep my code even if not referenced */
}> FLASH
```

在源文件中把 `CalSum ()` 函数放在 `0x10016000` 的地址：

```
void __attribute__((section(".myCodeSection"))) CalSum(void)
{
    ...
}
```

#### 5) 如何把一个文件放在 `Flash` 中的指定位置？

这里有两种方法可以实现这个功能。

- **方法一：把 `Flash` 划分成两块，把文件放在其中一块。**

例如把 `test2.c` 的文件放在 `0x10020000` 开始的地址，可以如下这么做。

首先把 Flash 划分成两块:

```
MEMORY
{
    FLASH(RX) : ORIGIN = 0x10001000, LENGTH = 0x20000

    /* Add a new Flash area */
    FLASH_1(RX) : ORIGIN = 0x10020000, LENGTH = 0x10000

    SRAM(!RX) : ORIGIN = 0x20000000, LENGTH = 0x4000
}
```

把源文件 test2.c 放在 Flash\_1 区域:

```
/* List files which are located in Flash_1 area */
.romCode :
{
    *test2.o (.text .text*);
} > FLASH_1
```

- 方法二: 先把源文件从 text 段 exclude 出来, 在最后定义一个代码段, 把源文件放置在那里。

例如把 test2.c 的文件放在 0x10020000 开始的地址, 可以如下这么做。

首次把源文件从 text 中 exclude 出来:

```
.text :
{
    ...
    *(EXCLUDE_FILE( *test2.o) .text.* .gnu.linkonce.t.*);
    ...
} > FLASH
```

添加到自己定义的代码段:

```
.romCode 0x10020000:
{
    *test2.o (.text .text*);
} > FLASH
```

采用这个方法需要确保 0x10020000 这个地址没有和别的代码重叠。

#### 4.4.11 查看程序运行周期数

在 DAVE 中没有寄存器可以直接查看某条语句或者某个函数的执行时间，但是我们可以利用 ARM Cortex-M 核自带的 System Tick 来实现这个功能。System Tick 是以 CPU Clock 作为它的源时钟，因此 1 个 System Tick 的 count 就表示一个机器周期。

例如需要计算 CMSIS-DSP 中 arm\_sin\_f32() 的执行时间，则可以用下面程序。简单的说就是在这个 API 的前把 System tick 清 0，在 API 执行完后读取 System tick 计数值，读取的计数值就是这个 API 的执行周期数。

```
uint32_t DURATION_TIME;
float result1;

SysTick->CTRL = 0; /* Disable SysTick */
SysTick->LOAD = 0xfffff; /* Count down from maximum value */
SysTick->CTRL = 0x5; /* Enable SysTick and use processor clock */
SysTick->VAL = 0; /* Clear current value to 0 */

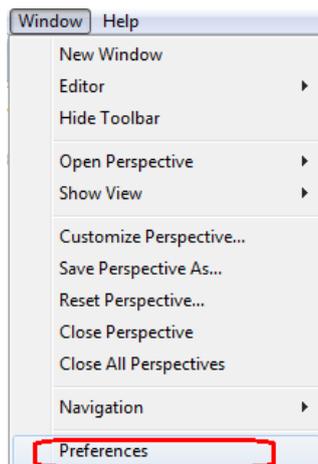
result1 = arm_sin_f32(0.5f);

DURATION_TIME = SysTick->VAL; /* Get duration time */
SysTick->CTRL = 0; /* Disable SysTick */
```

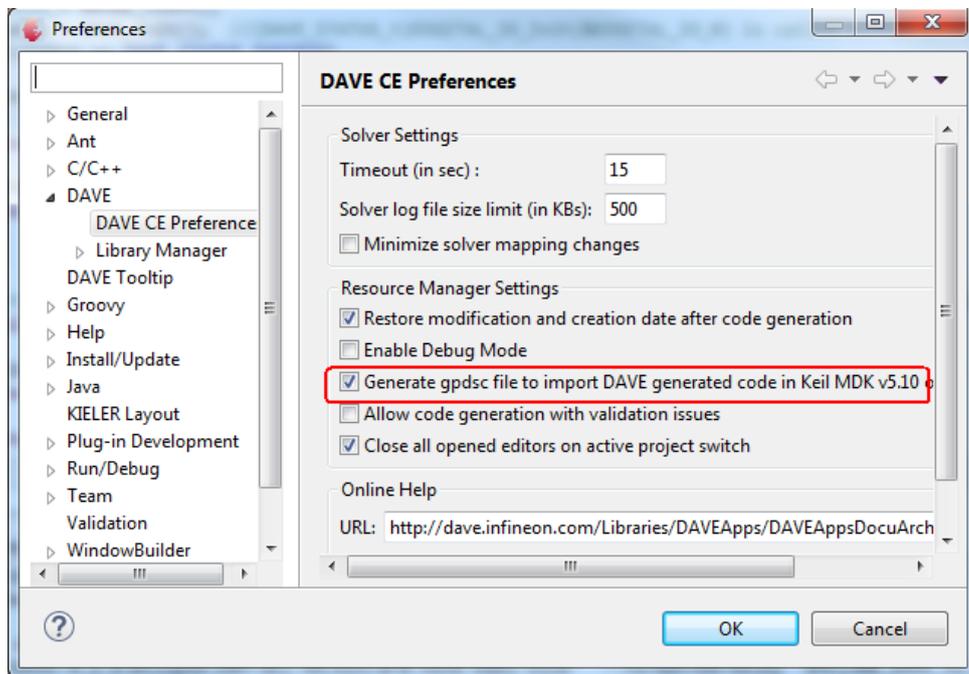
#### 4.4.12 DAVE 工程转 Keil 工程流程

##### DAVE4 APP 工程转 Keil MDK 步骤

- 1) 打开或新建一个 DAVE 工程
- 2) 点击 Window|Preferences

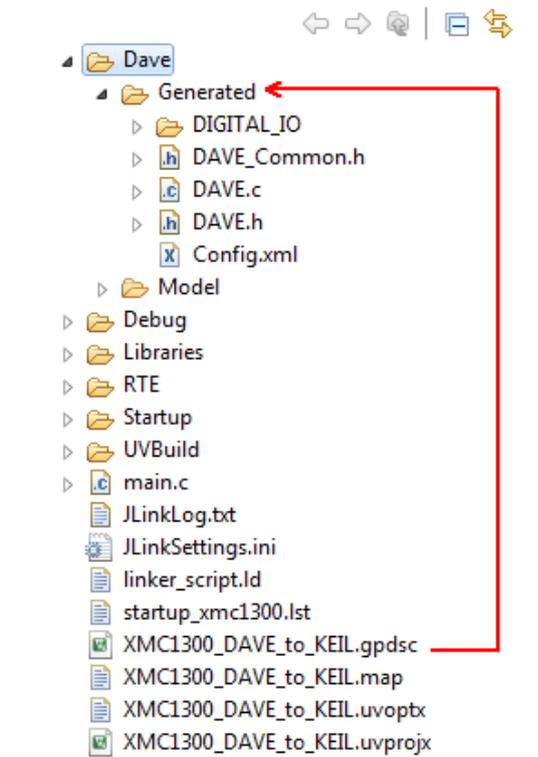


3) 选择 DAVE CE Preference, 勾选下面选项



4) 点击 , 重新生成代码

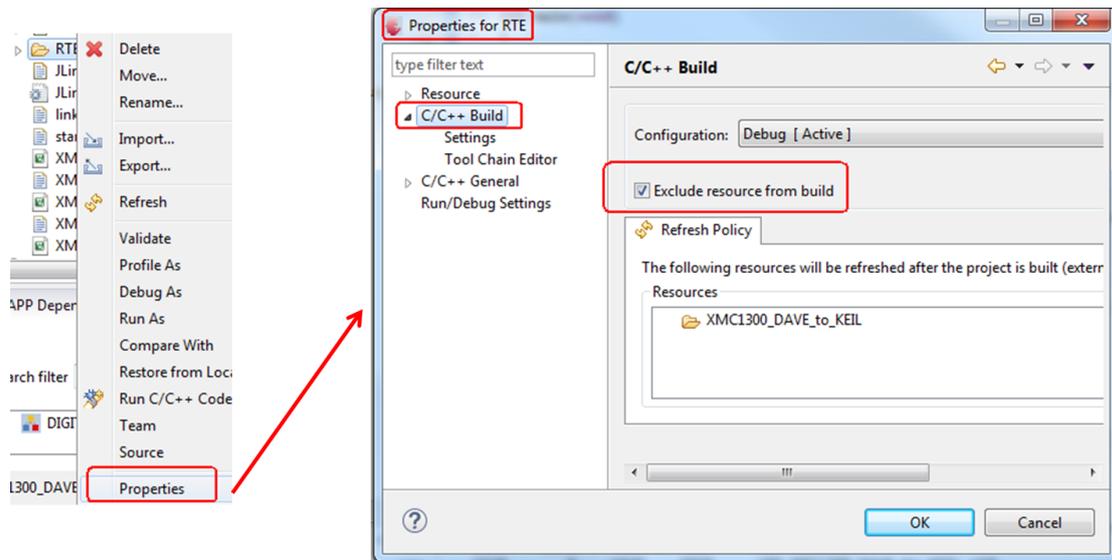
5) 重新生成代码后, 在工程目录中会增加一个 gpdsc 文件



注意:

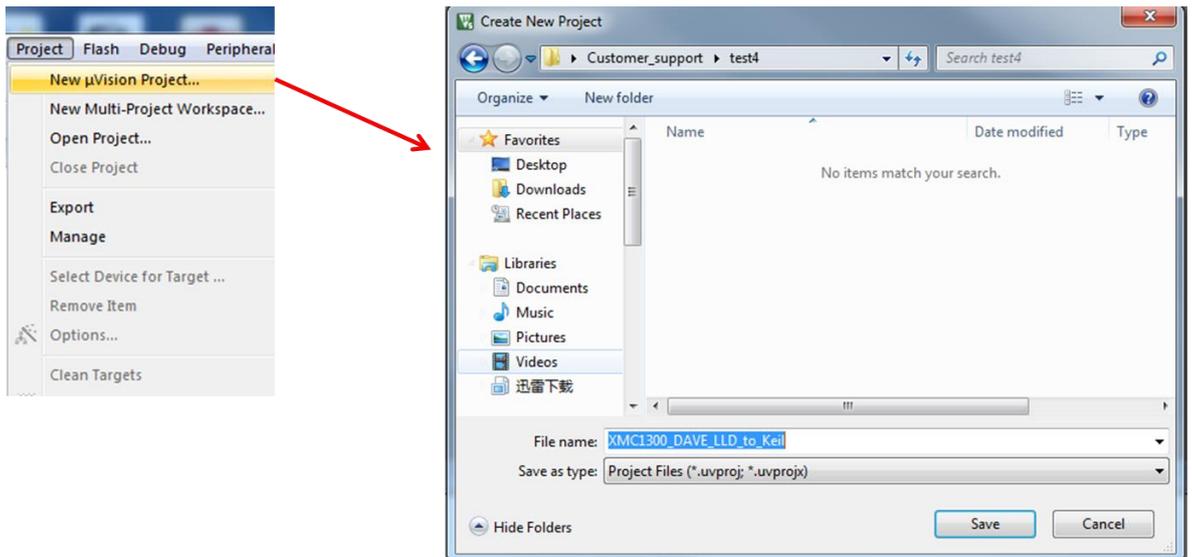
- 只有 DAVE APP (DAVE CE) 工程可以通过这个方式转成 Keil
- *gpdsc* 仅仅描述生成的代码文件的结构, 即 *Generated* 文件夹下的这些文件

在重新生成代码的过程中, 生成了 RTE 文件夹, 如果需要继续在 DAVE 中编译, 则需要把这个文件夹 exclude.

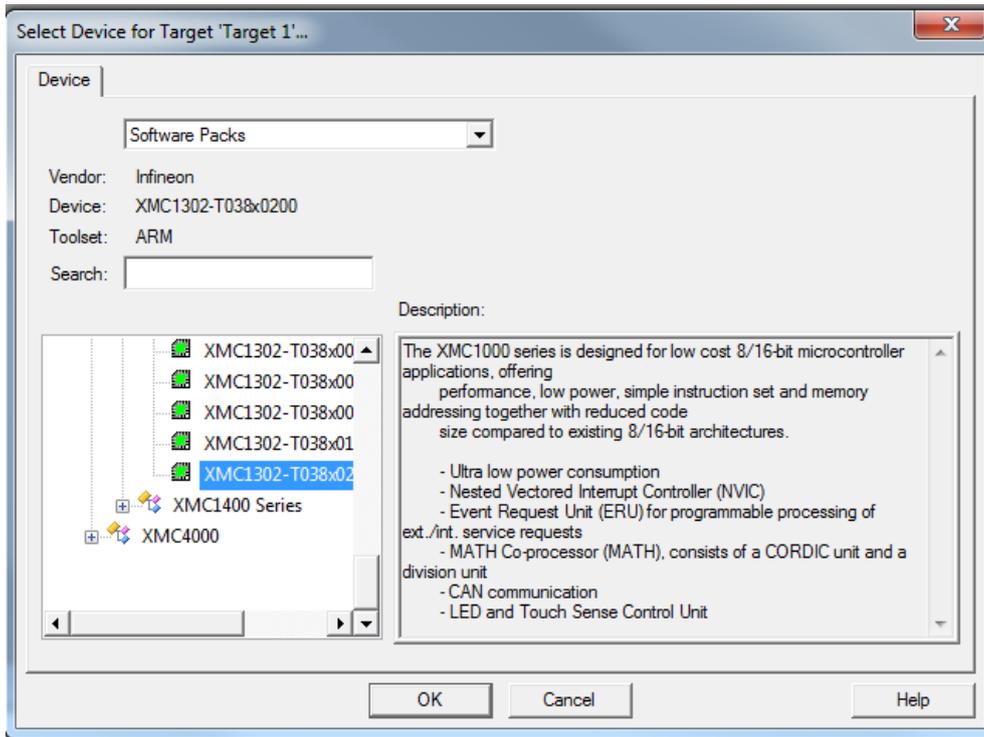


## DAVE4 LLD 工程 转 Keil MDK 步骤

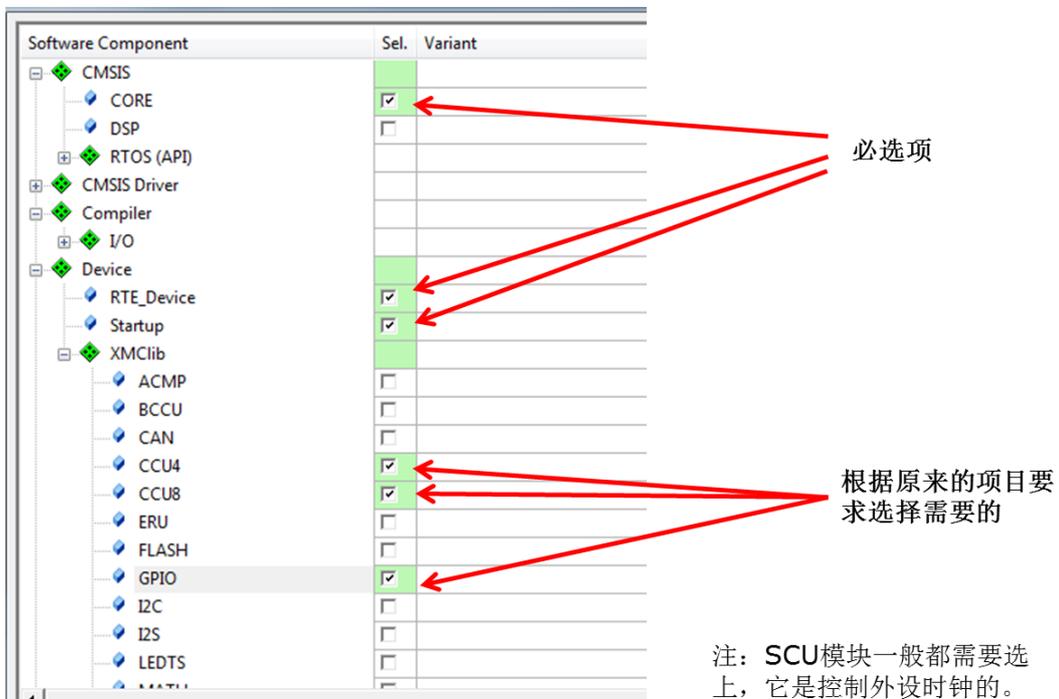
1) 新建一个 Keil 工程



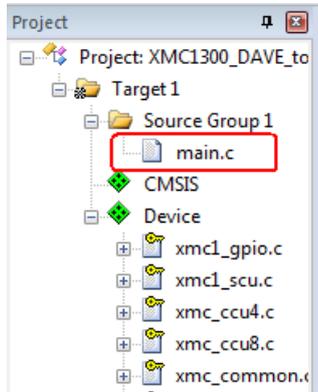
## 2) 选择对应芯片



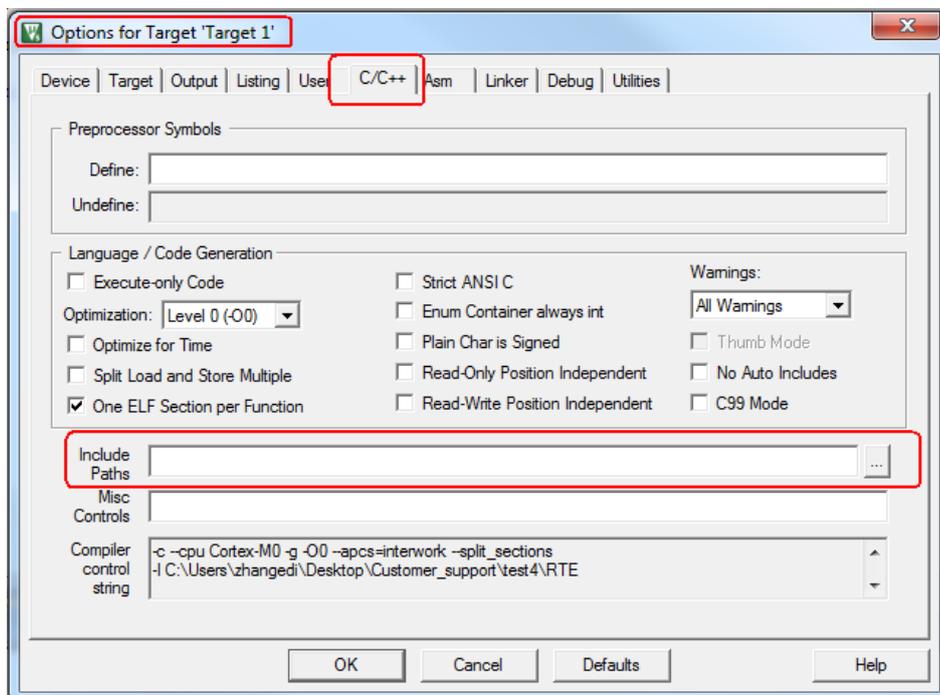
## 3) 选择 CMSIS 组件



4) 添加用户应用文件



5) 设置需要的预定义路径



6) 选择 debugger 后, 就可以在 Keil 环境下调试了



## 4.2 Keil MDK

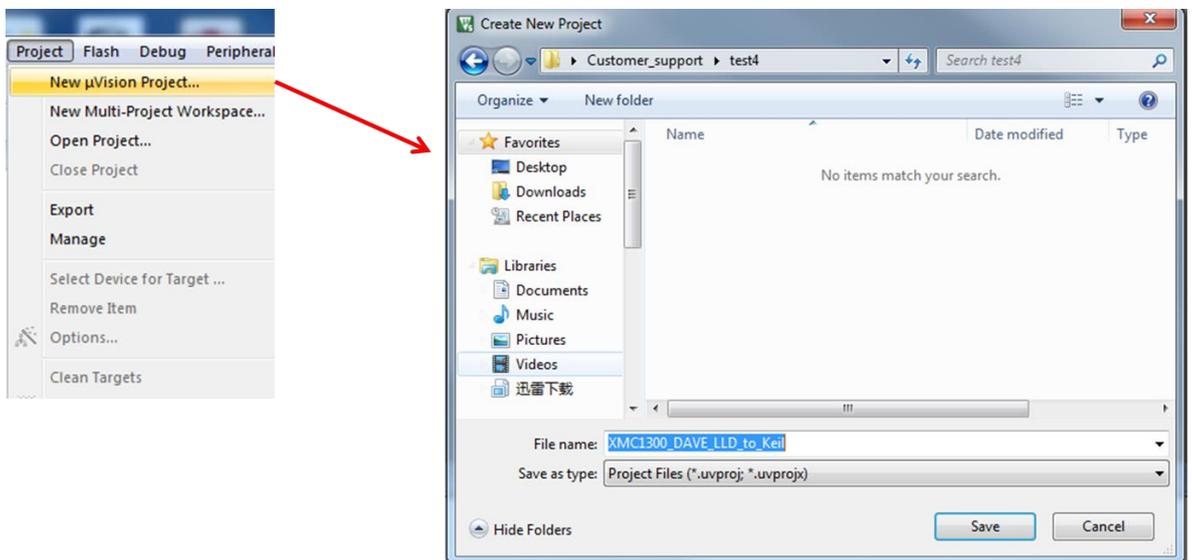
### 4.2.1 下载

可以从下面地址下载最新版的 Keil MDK 版本:

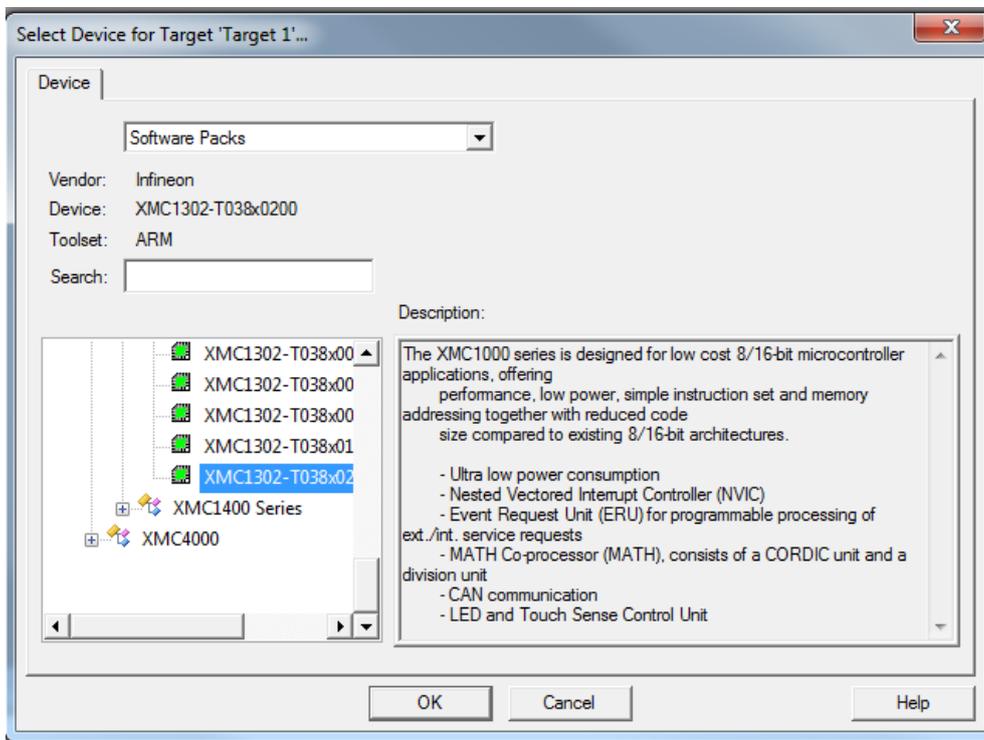
<https://www.keil.com>

### 4.2.2 新建工程

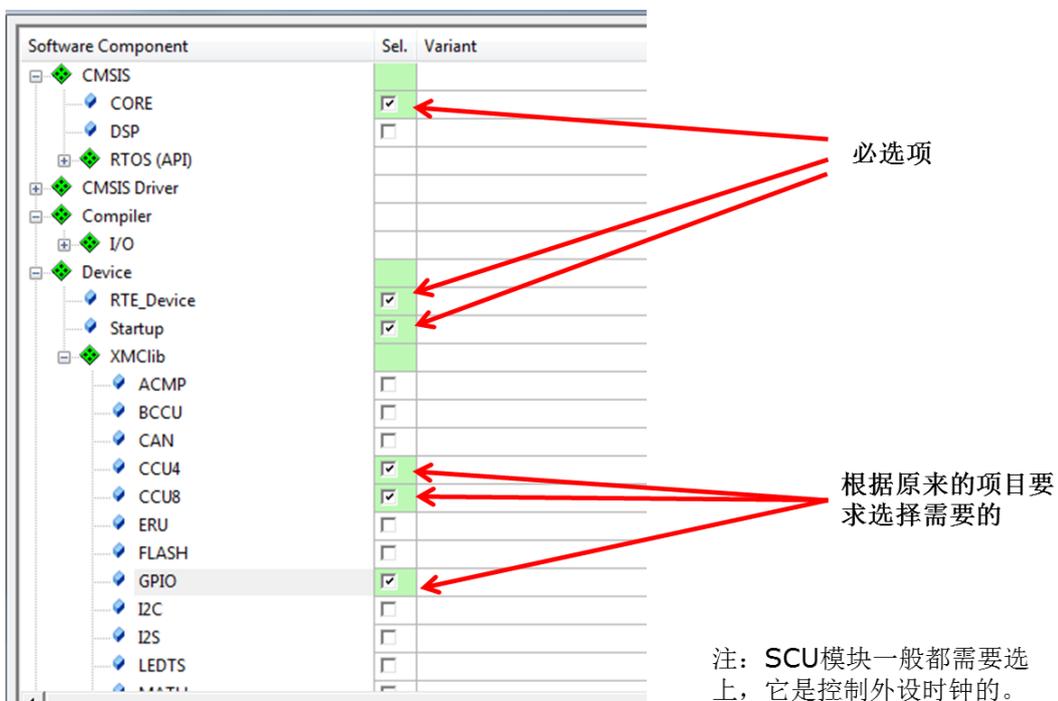
1) 新建一个 Keil 工程



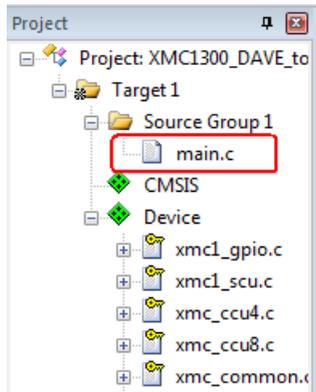
## 2) 选择对应芯片



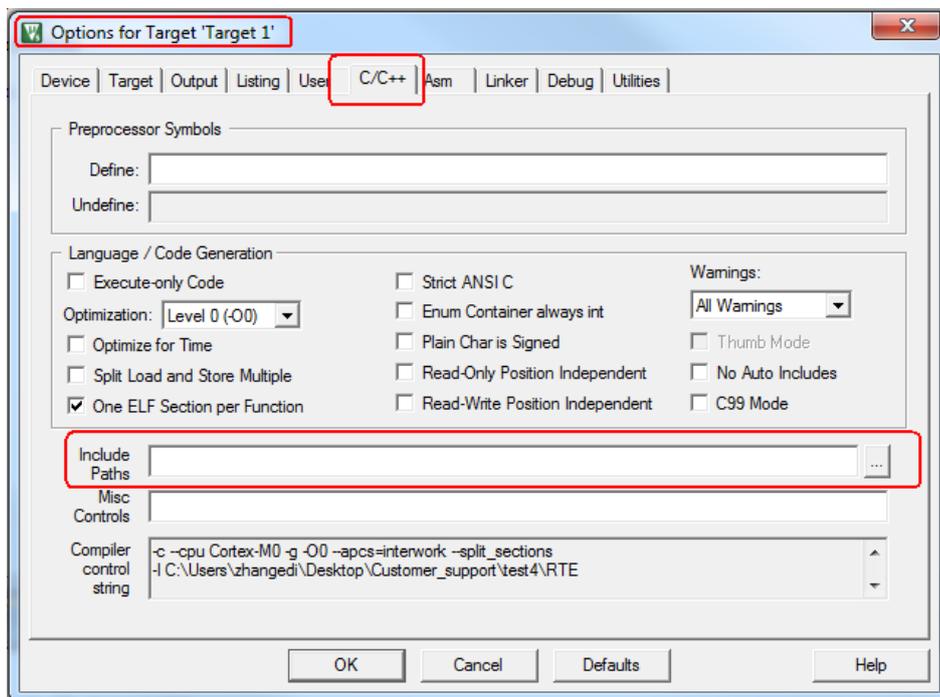
## 3) 选择 CMSIS 组件



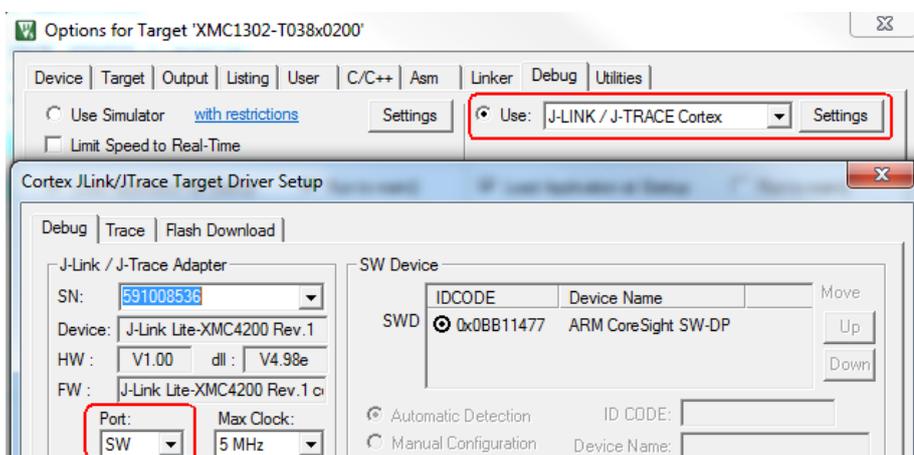
4) 添加用户应用文件



5) 设置需要的预定义路径

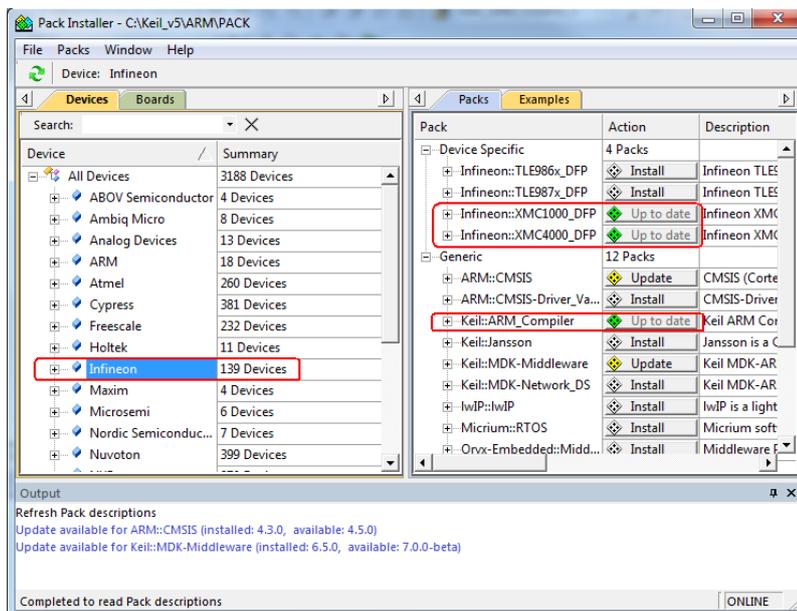


6) 选择 debugger 后，就可以在 Keil 环境下调试了

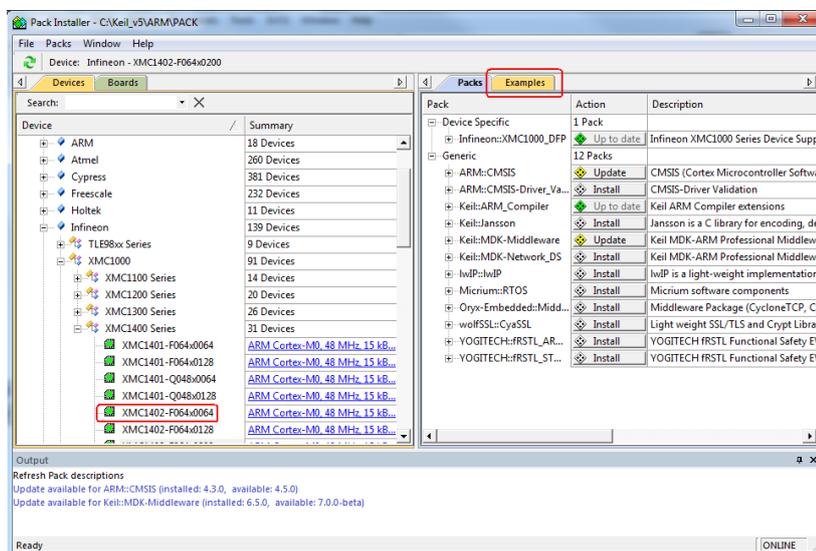


## 4.2.3 打开自带例程

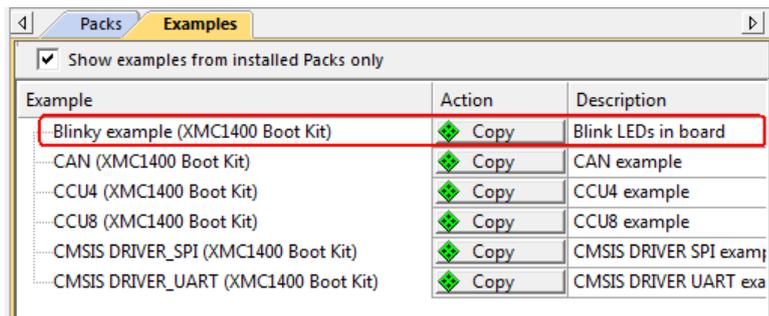
- 1) 打开 Keil
- 2) 点击  ，打开 Pack Installer
- 3) 选择 Infineon
- 4) 安装 XMC1000\_DFP, XMC4000\_DFP, Keil Compiler 等（如已安装，则跳过）



- 5) 选择芯片型号，例如 XMC1402-F064x0200
- 6) 点 Example



7) 选择其中一个 Example



8) 点击 Copy

9) 编译

10) 下载和运行

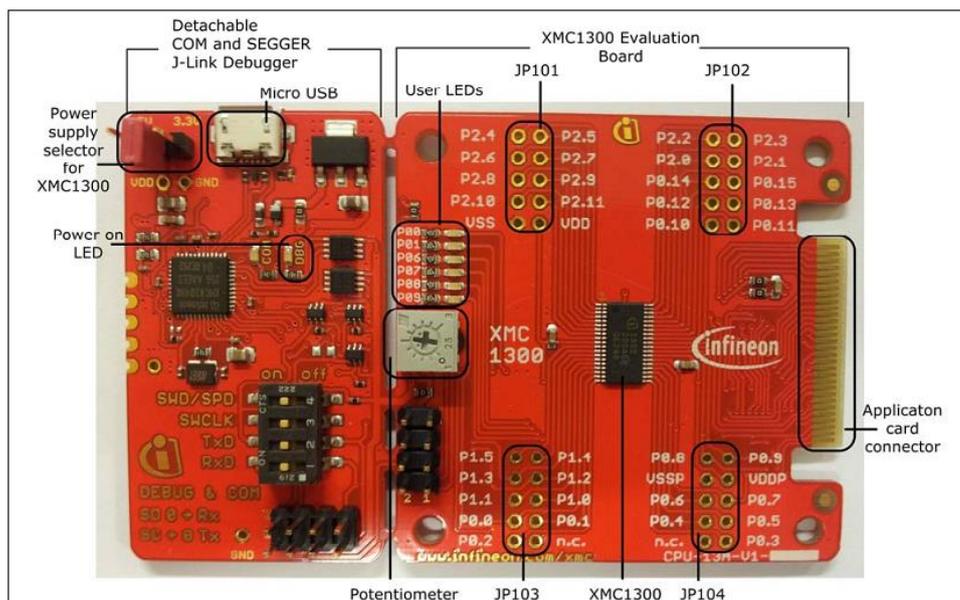
## 5 调试工具

### 5.1 英飞凌板载 J-Link

#### 5.1.1 XMC1000 板载 J-Link

XMC1000 系列的开发套件在板上集成了 J-Link。这个 J-Link 的功能完全兼容标准的 SEGGER J-Link，而且还能模拟出一个虚拟串口。它的引脚配置如下：

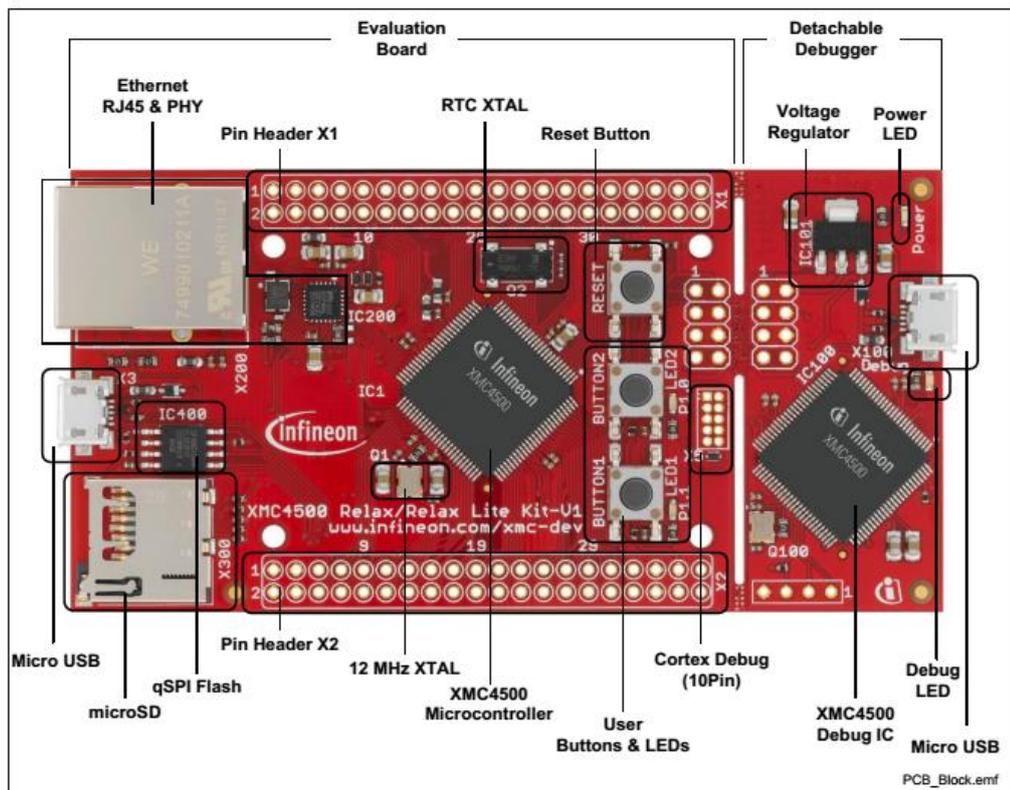
Pin	Signal Name	XMC1300 Signal	Description
1	SWCLK	P0.15	SWD clock signal
2	SWD (SPD)	P0.14	SWD/SPD signal
3	VDDP	VDDP	+5V
4	GND	VSSP	Ground
5	GND	VSSP	Ground
6	VDDP	VDDP	+5V
7	PC_TXD	P1.3	UART data received
8	PC_RXD	P1.2	UART data transmit



### 5.1.2 XMC4000 板载 J-Link

XMC4000 系列的开发套件在板上集成了 J-Link。这个 J-Link 的功能完全兼容标准的 SEGGER J-Link，而且还能模拟出一个虚拟串口。相比 XMC1000 的板载 J-Link，它增加了 SWV 和 Reset 引脚。它的引脚配置如下：

Pin	Signal Name	XMC4500 Signal	Description
1	SWD	TMS	SWD signal
2	VCC	VDDP	Power signal
3	SWCLK	TCK	SWD clock signal
4	Reset	PORST#	Reset signal
5	SWV	-	SWV trace data output
6	GND	VSS	Ground
7	PC_RXD	P0.1	UART data transmit
8	PC_TXD	P0.0	UART data receive



## 5.2 Segger J-link

SEGGER 标准的 20pin JTAG 接口如下:

Pin	Signal Name	Description
1	VTref	Target reference voltage
2	NC	No connection
3	nTRST	JTAG Reset
5	TDI	JTAG data input of target CPU
7	TMS	JTAG mode set input of target CPU
9	TCK	JTAG clock signal to target CPU
11	NC	No connection
13	TDO	JTAG data output from target CPU
15	RESET	Target CPU reset signal
17	NC	No connection
19	5V-Supply	This pin can be used to supply power to the target hardware
4,6,8,10,12,14,16,18,20	GND	Ground

可以查看下面文档:

[https://www.segger.com/admin/uploads/productDocs/UM08001\\_JLink.pdf](https://www.segger.com/admin/uploads/productDocs/UM08001_JLink.pdf)

## 5.3 XMC-Link

XMC-Link 是英飞凌公司开发的 J-Link 仿真器。除了兼容标准的 SEGGER J-Link 功能之外，它还能模拟出一个虚拟串口，另外还带隔离的功能。

它的 10-pin 接口引脚分配如下:

Pin	Function	XMC1000 Connection	XMC4000 Connection
1	VCC	Power supply 2.5V-5.5V(VDD)	Power supply VDDP 3.3V (VDDP)
2	SWIO/TMS	Serial Wire Data (P0.14   P1.3)	Serial Wire Data, JTAG-TMS
3	GND	Ground (VSS)	Ground (VSS)
4	SWCLK/TCK	Serial Wire Clock (P0.15   P1.2)	Serial Wire Clock, JTAG-TCK(TCK)
5	GND	Ground (VSS)	Ground (VSS)
6	SWO/TDO	NC	Serial Wire Output, JTAG-TDO (P2.1) (optional)
7	KEY	NC	NC
8	TDI	NC	JTAG-TDI (P0.7)(optional)
9	GNDDetect	NC	NC
10	RESET#	NC	PORST# (mandatory)

它的 8-pin 接口引脚分配如下：

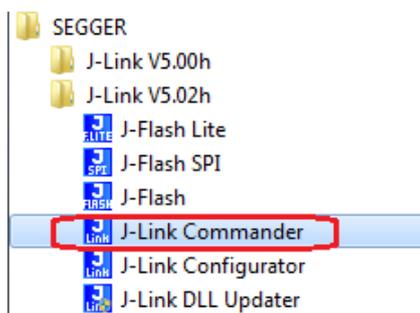
Pin	Function	XMC1000 Connection (Pin name)
1	SWCLK	Serial Wire Clock (P0.15   P1.2)
2	SWD	Serial Wire Data (P0.14   P1.3)
3	+	Power supply 2.5 V – 5.5 V (VDD)
4	GND	Ground (VSS)
5	GND	Ground (VSS)
6	+	Power supply 2.5 V – 5.5 V (VDD)
7	TX (PC-TX)	Transmissstion line of PC/laptop, receive line of XMC™ device (optional)
8	RX (PC-RX)	Receive line of PC/laptop, transmission line of of XMC™ device (optional)



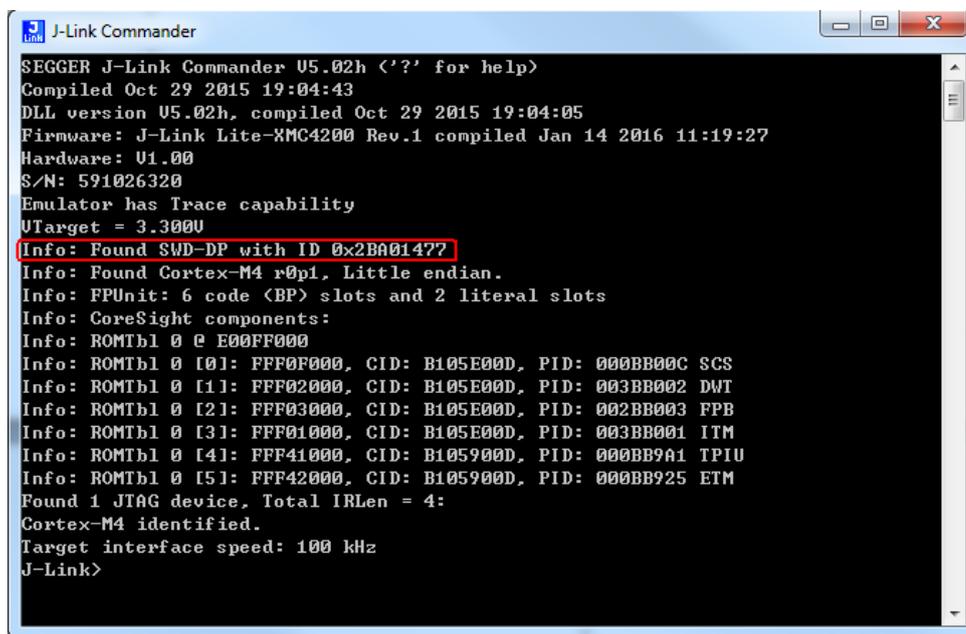
## 5.4 J-Link 命令行

J-Link 命令行能实现一些实用的功能，例如擦除 Flash 和修改 BMI。

可以在开始菜单，打开最新的 SEGGER J-Link Commander



打开会显示以下界面，如果可以读到 ID，则说明 J-Link 连接正常。



```
J-Link Commander
SEGGER J-Link Commander V5.02h ('?' for help)
Compiled Oct 29 2015 19:04:43
DLL version V5.02h, compiled Oct 29 2015 19:04:05
Firmware: J-Link Lite-XMC4200 Rev.1 compiled Jan 14 2016 11:19:27
Hardware: V1.00
S/N: 591026320
Emulator has Trace capability
UTarget = 3.300U
Info: Found SWD-DP with ID 0x2BA01477
Info: Found Cortex-M4 r0p1, Little endian.
Info: FPUnit: 6 code <BP> slots and 2 literal slots
Info: CoreSight components:
Info: ROMTbl 0 @ E00FF000
Info: ROMTbl 0 [0]: FFF0F000, CID: B105E00D, PID: 000BB00C SCS
Info: ROMTbl 0 [1]: FFF02000, CID: B105E00D, PID: 003BB002 DWT
Info: ROMTbl 0 [2]: FFF03000, CID: B105E00D, PID: 002BB003 FPB
Info: ROMTbl 0 [3]: FFF01000, CID: B105E00D, PID: 003BB001 ITM
Info: ROMTbl 0 [4]: FFF41000, CID: B105900D, PID: 000BB9A1 TPIU
Info: ROMTbl 0 [5]: FFF42000, CID: B105900D, PID: 000BB925 ETM
Found 1 JTAG device, Total IRLen = 4:
Cortex-M4 identified.
Target interface speed: 100 kHz
J-Link>
```

### 5.4.1 擦除 Flash

输入下面命令可以整片擦除 Flash。

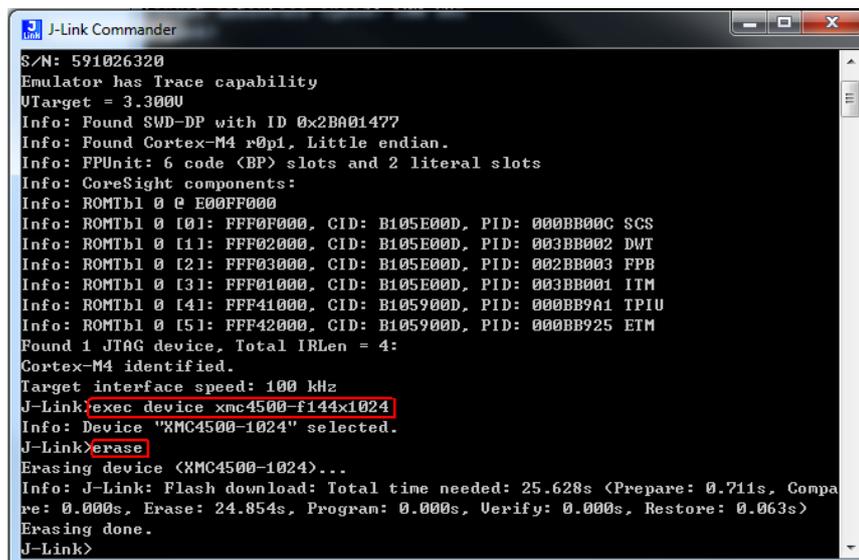
```
exec device device_name
```

```
erase
```

例如擦除 XMC4500-F144x1024，则输入：

```
exec device xmc4500-f144x1024
```

```
erase
```

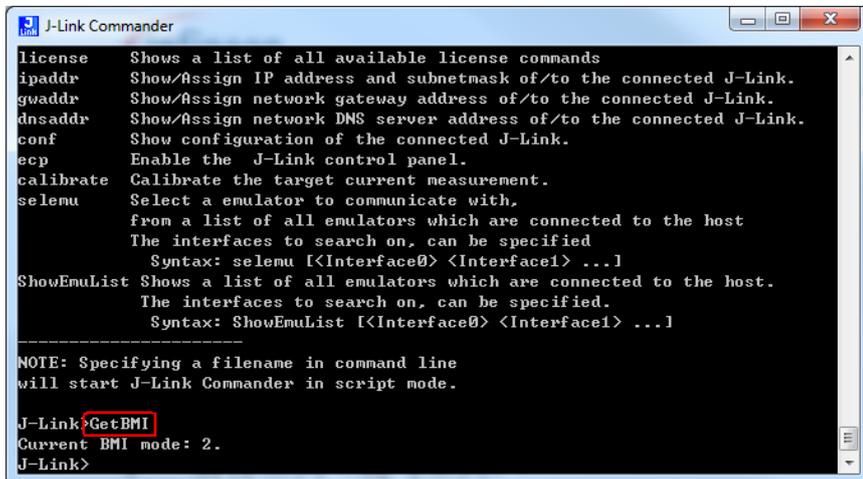


```
J-Link Commander
S/N: 591026320
Emulator has Trace capability
UTarget = 3.300U
Info: Found SWD-DP with ID 0x2BA01477
Info: Found Cortex-M4 r0p1, Little endian.
Info: FPUnit: 6 code <BP> slots and 2 literal slots
Info: CoreSight components:
Info: ROMTbl 0 @ E00FF000
Info: ROMTbl 0 [0]: FFF0F000, CID: B105E00D, PID: 000BB00C SCS
Info: ROMTbl 0 [1]: FFF02000, CID: B105E00D, PID: 003BB002 DWT
Info: ROMTbl 0 [2]: FFF03000, CID: B105E00D, PID: 002BB003 FPB
Info: ROMTbl 0 [3]: FFF01000, CID: B105E00D, PID: 003BB001 ITM
Info: ROMTbl 0 [4]: FFF41000, CID: B105900D, PID: 000BB9A1 TPIU
Info: ROMTbl 0 [5]: FFF42000, CID: B105900D, PID: 000BB925 ETM
Found 1 JTAG device, Total IRLen = 4:
Cortex-M4 identified.
Target interface speed: 100 kHz
J-Link>exec device xmc4500-f144x1024
Info: Device "XMC4500-1024" selected.
J-Link>erase
Erasing device <XMC4500-1024>...
Info: J-Link: Flash download: Total time needed: 25.628s <Prepare: 0.711s, Compars: 0.000s, Erase: 24.854s, Program: 0.000s, Verify: 0.000s, Restore: 0.063s>
Erasing done.
J-Link>
```

## 5.4.2 修改 BMI

J-Link Commander 还可以查看和修改 BMI。

输入“GetBMI”，可以查看当前 BMI 的模式。

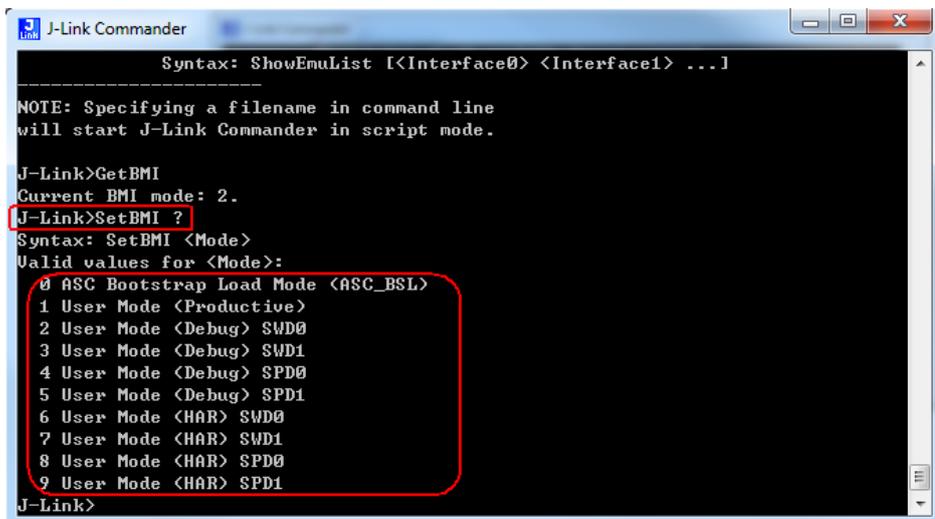


```
J-Link Commander
license      Shows a list of all available license commands
ipaddr      Show/Assign IP address and subnetmask of/to the connected J-Link.
gwaddr      Show/Assign network gateway address of/to the connected J-Link.
dnsaddr     Show/Assign network DNS server address of/to the connected J-Link.
conf        Show configuration of the connected J-Link.
ecp         Enable the J-Link control panel.
calibrate   Calibrate the target current measurement.
selemu      Select an emulator to communicate with,
             from a list of all emulators which are connected to the host
             The interfaces to search on, can be specified
             Syntax: selemu [<Interface0> <Interface1> ...]
ShowEmuList Shows a list of all emulators which are connected to the host.
             The interfaces to search on, can be specified.
             Syntax: ShowEmuList [<Interface0> <Interface1> ...]

NOTE: Specifying a filename in command line
will start J-Link Commander in script mode.

J-Link>GetBMI
Current BMI mode: 2.
J-Link>
```

上面显示 BMI 模式是 2，输入“SetBMI ?”，可以查看 2 的含义。



```
Syntax: ShowEmuList [<Interface0> <Interface1> ...]

NOTE: Specifying a filename in command line
will start J-Link Commander in script mode.

J-Link>GetBMI
Current BMI mode: 2.
J-Link>SetBMI ?
Syntax: SetBMI <Mode>
Valid values for <Mode>:
0 ASC Bootstrap Load Mode (ASC_BSL)
1 User Mode (Productive)
2 User Mode (Debug) SWD0
3 User Mode (Debug) SWD1
4 User Mode (Debug) SPD0
5 User Mode (Debug) SPD1
6 User Mode (HAR) SWD0
7 User Mode (HAR) SWD1
8 User Mode (HAR) SPD0
9 User Mode (HAR) SPD1

J-Link>
```

输入“SetBMI 0”，可以切换到 ASC Bootstrap Load Mode。



```
Cortex-M0 identified.
Target interface speed: 100 kHz
J-Link>SetBMI 0
Setting BMI mode 0...O.K.
J-Link>
```

注意：切换到 ASC\_BSL 模式后就不能再连接 J-Link 了，需要在 DAVE 中把 BMI 模式改成 User Mode (SWD)后才能继续用 J-Link 调试。

## 6 编程工具（烧写工具）

### 6.1 MemTool

MemTool 是能通过串口烧写 XMC MCU 片内 Flash 的工具。MemTool 的下载地址可以查看第一章。

#### 6.1.1 Boot 模式

在了解 MemTool 用法之前，首先要先了解 MCU 的 Boot 模式。

##### XMC1000 Boot 模式：

XMC1000 系列有很多种 Boot 模式，包括 ASC\_BSL (Full duplex), User Mode (SWD), User Mode (SPD), ASC\_BSL (Half duplex) 和 SSC\_BSL。这些模式所用的硬件引脚如下：

		ASC_BSL (Full duplex)	User Mode SWD0/1	User Mode SPD0/1	ASC_BSL (Half duplex)	SSC_BSL
XMC1000	Channel 0 P0.13 P0.14 P0.15	-	-	-	-	/CS
		RXD	SWDIO	SPD	RxD-TxD(Half duplex)	SCLK
		TXD	SWDCLK	-	-	MRST-MTSR
	Channel 1 P1.3 P1.2	RXD	SWDIO	SPD	RxD-TxD(Half duplex)	
		TXD	SWDCLK	-	-	

芯片出厂时的默认 Boot 模式是 ASC\_BSL (Full duplex). 在这个模式下可以使用 MemTool 通过串口线对 MCU 进行烧写。

如果当前 MCU 的 Boot 模式是 user mode, 可以用 DAVE 把 BMI 模式设为 ASC\_BSL 模式（查看 4.4.6），然后再用 MemTool 烧写。

##### XMC4000 Boot 模式：

XMC4000 的 Boot mode 是由两个外部引脚 TCK 和 TMS 决定的。如果 TCK 和 TMS 这两个引脚悬空，则芯片上电后处在 Normal mode（即 TCK 和 TMS 引脚在芯片内部是弱下拉和弱上拉的）。而 UART 烧写所需模式是 ASC\_BSL 模式，因此在烧写之前需要把 TMS 下拉，把 Boot 模式配置成 ASC\_BSL 模式。

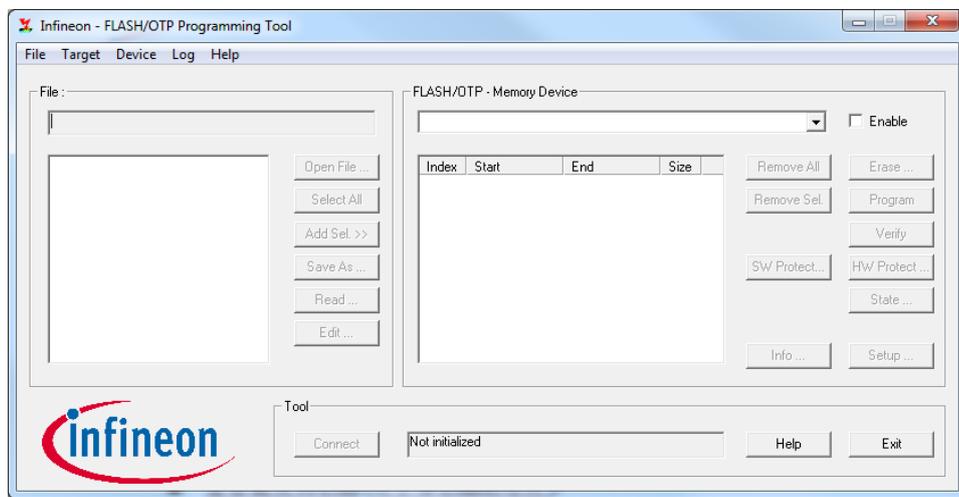
TCK	TMS	HWCON[1:0]	Boot mode
0	1	00 <sub>B</sub>	Normal
0	0	01 <sub>B</sub>	ASC BSL
1	1	10 <sub>B</sub>	BMI
1	0	11 <sub>B</sub>	CAN BSL

UART 烧写引脚是 P1.4 (U0C0\_DX0B)和 P1.5 (U0C0\_DOUT0)。这两个引脚是固定的。

## 6.1.2 烧写流程

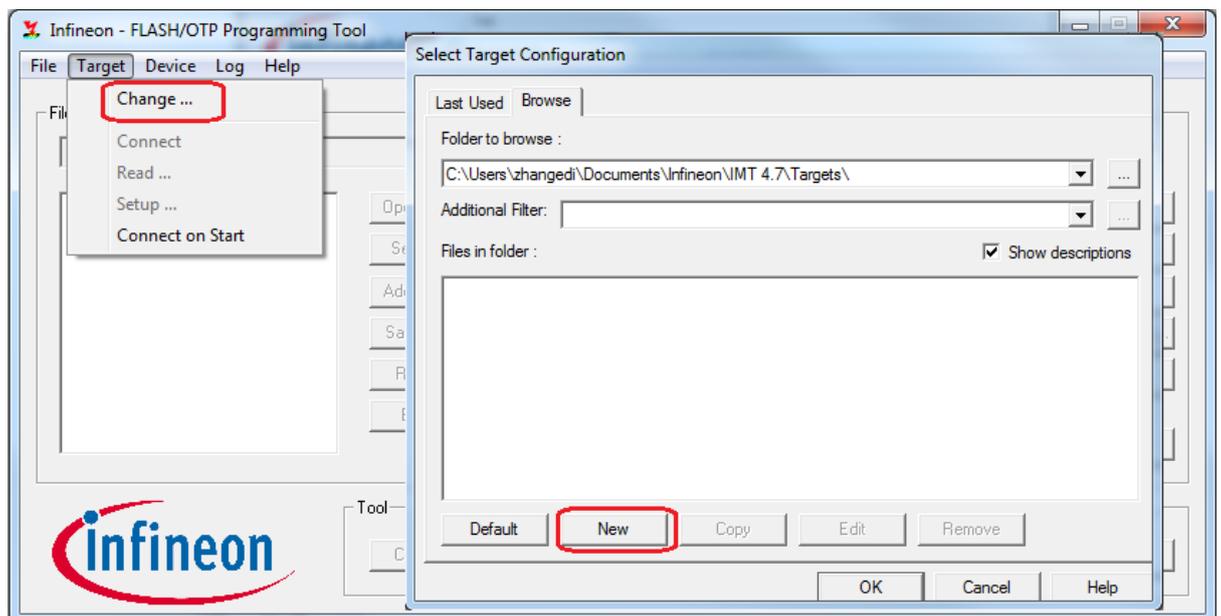
首先确保 MCU 是处在 ASC\_BSL 模式，串口连接正确（查看 6.1.1）。

### 1) 打开 MemTool

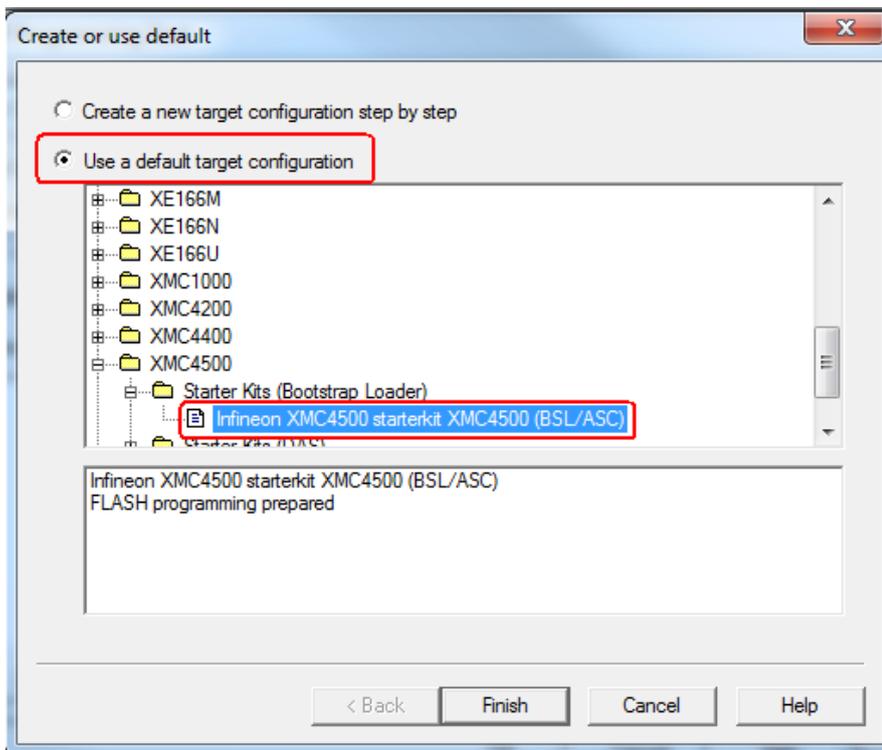


### 2) 如果第一次使用 MemTool，需要配置芯片信息步骤如下：

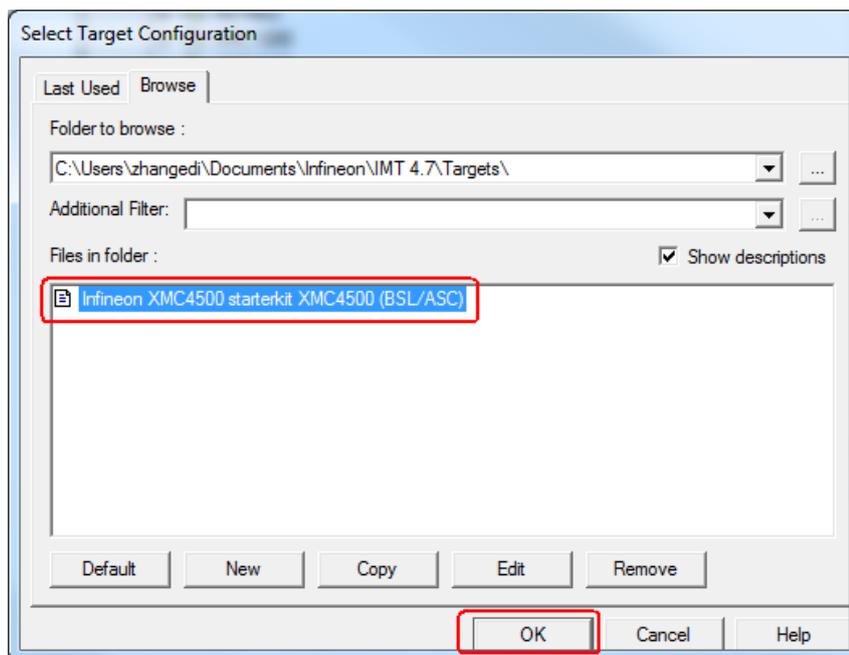
- 选择 Target -> Change
- 在弹出窗口选择 New



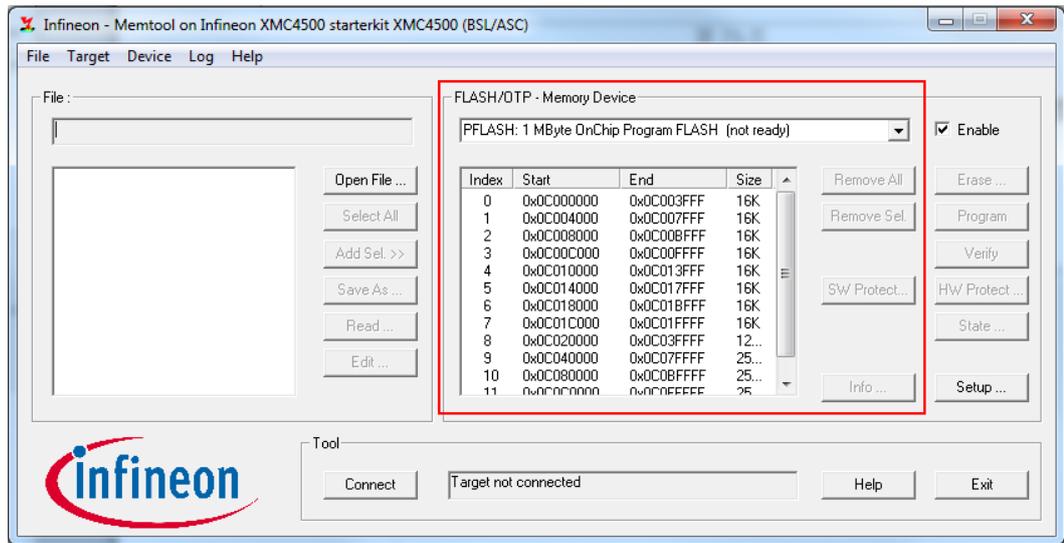
- 3) 选择”Use a default target configuration”，选择目标芯片(以 XMC4500 为例),点 Finish



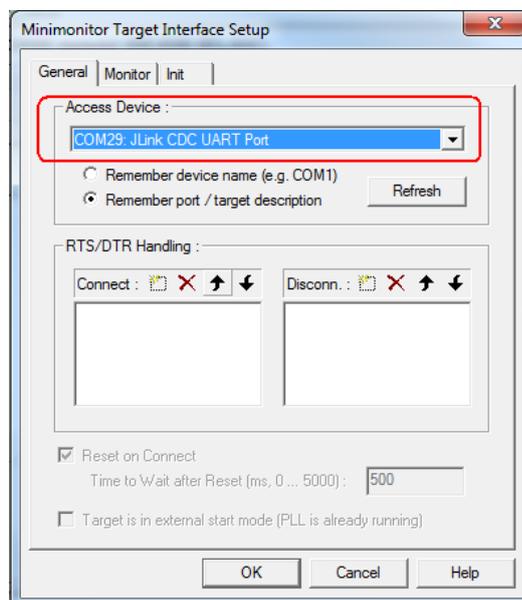
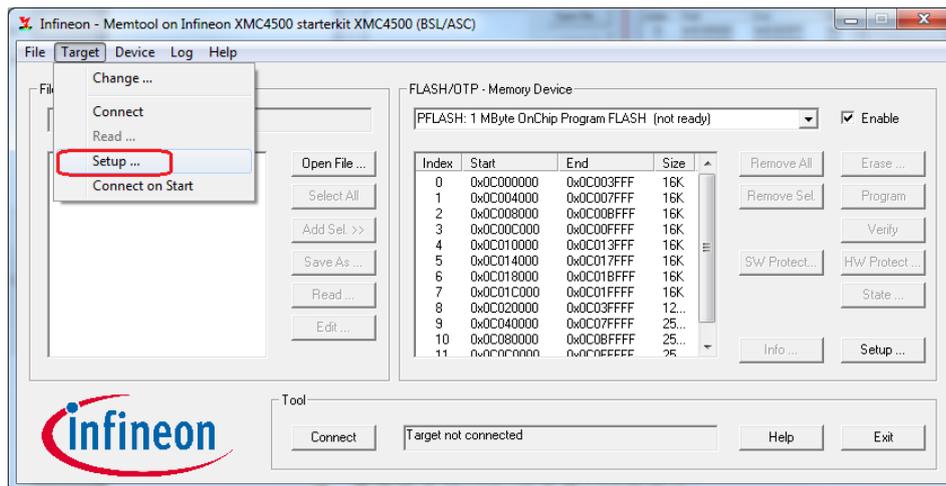
- 4) 在接下去弹出的窗口中，点击 Save
- 5) Save 之后，配置界面如下，点击 OK



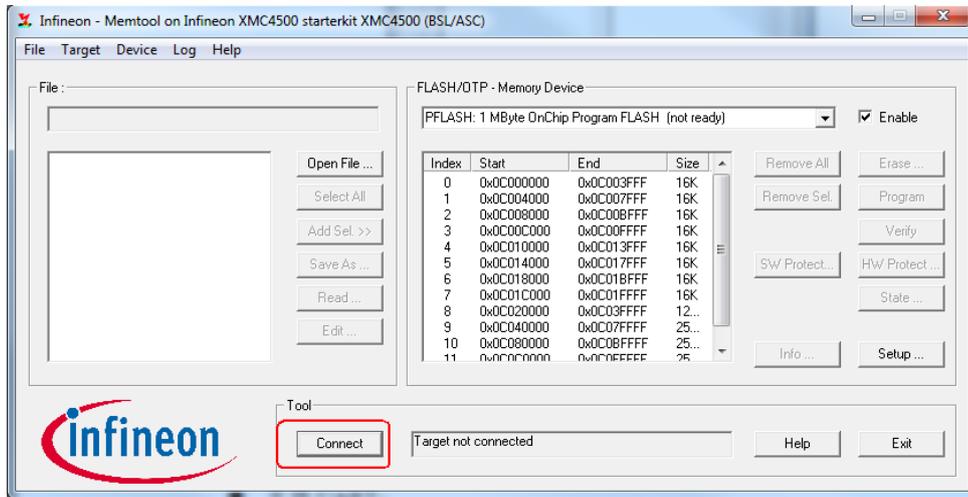
6) 下面列出了 XMC4500 各个 sector 的地址和容量



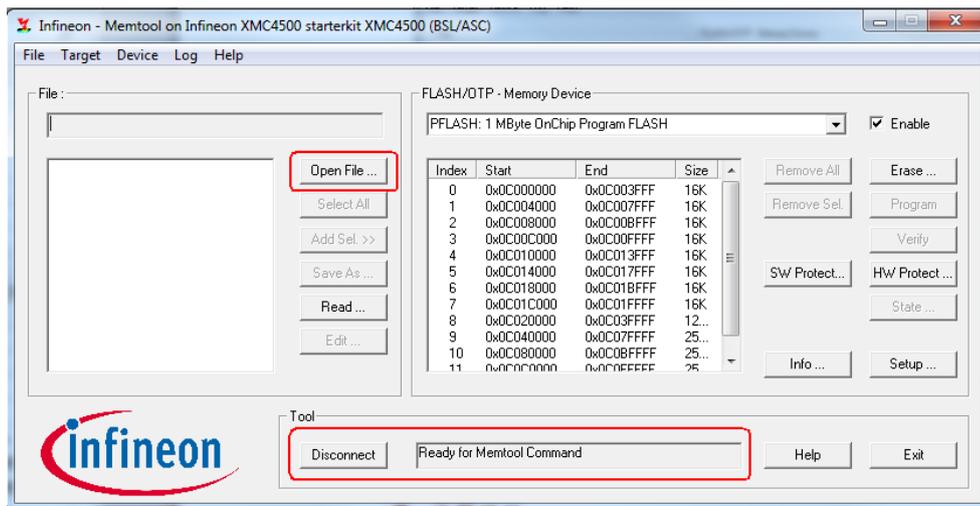
7) 点”Target->Setup...”, 选择 COM 口



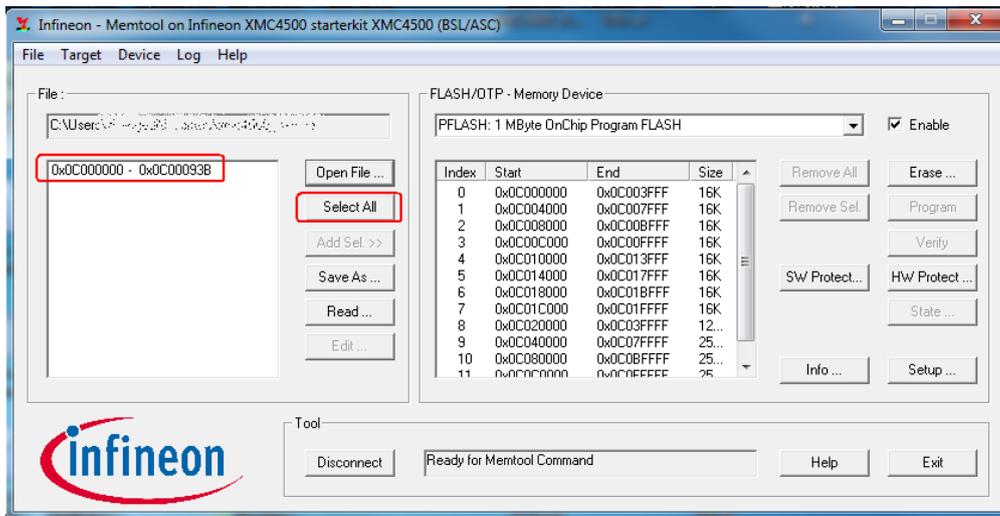
8) 点击 connect



9) 此时显示已经连接，接着点击 Open File，选择需要烧写的 Hex 文件

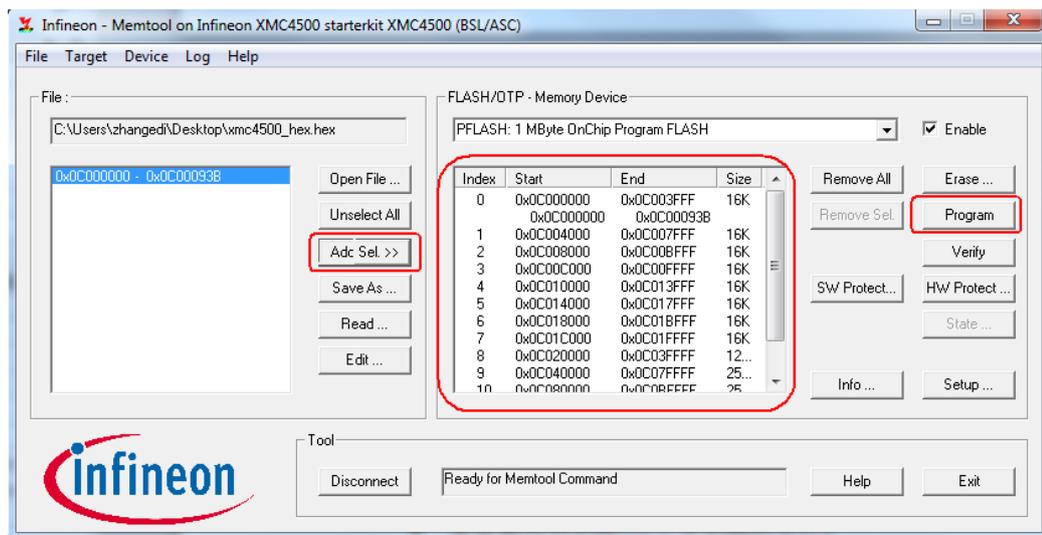


10) 点击 Select All



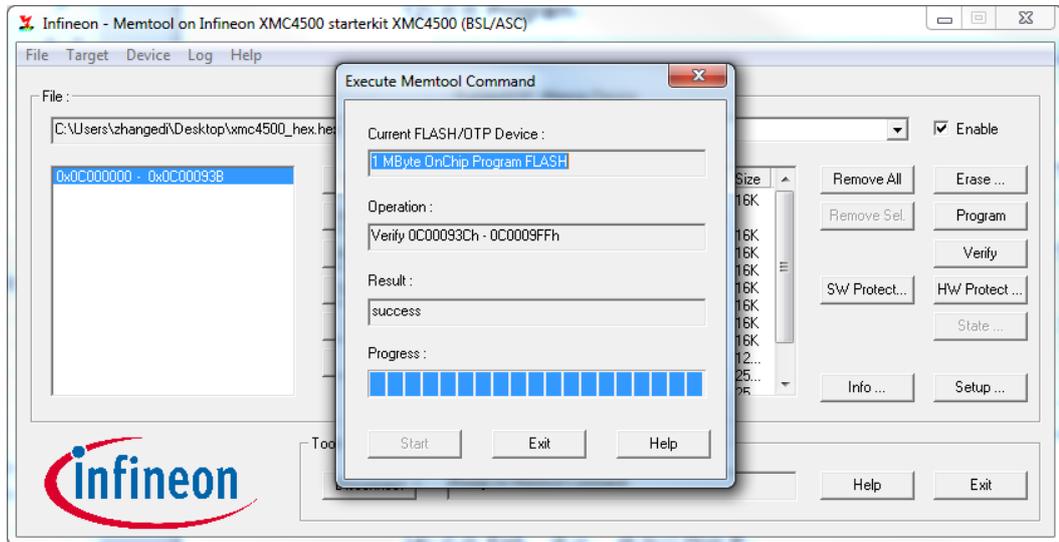
11) 点击 Add Sel, 右边地址会将左边内容添加至 Buffer 区

12) 点击 Program



13) 当提示 Success, 表示烧写完毕

14) 点击 **Exit**，退出，烧写过程结束

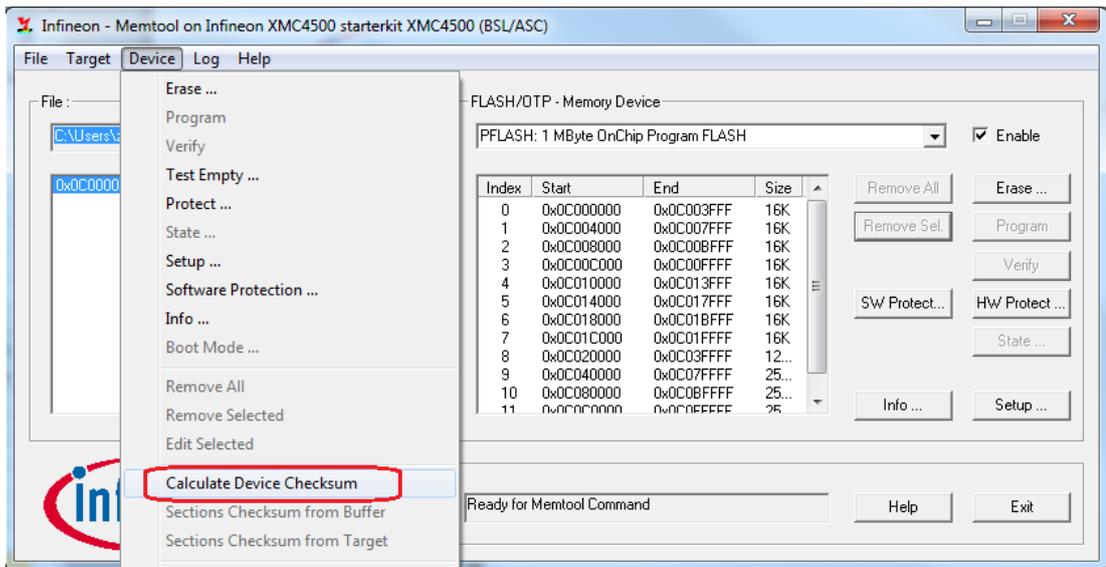


注意：默认情况下，**Program** 包括了烧写前对 **Flash** 的擦除和烧写后对烧写内容的校验。

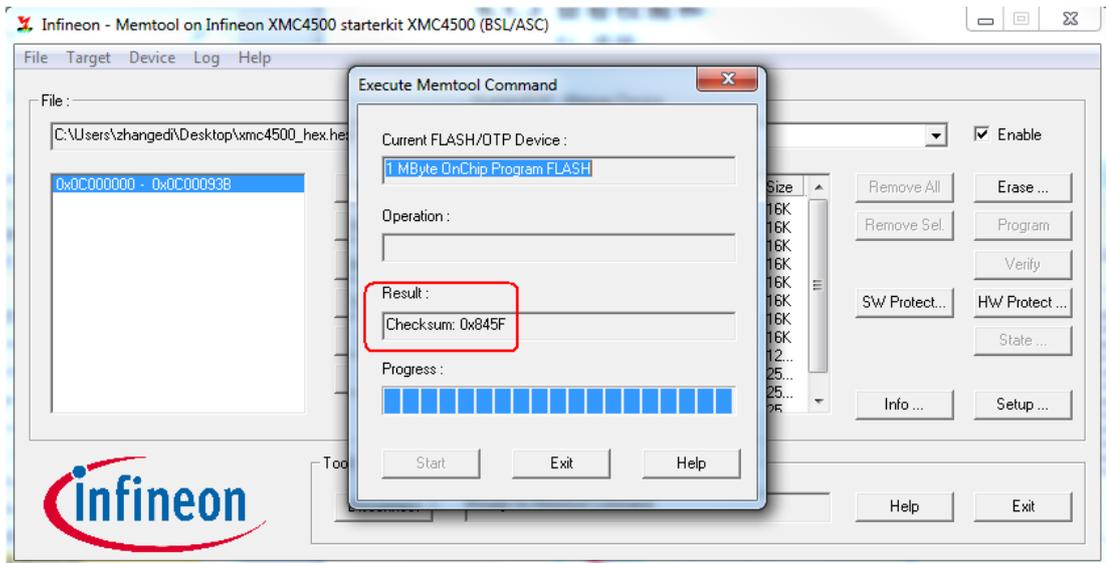
### 6.1.3 查看校验和

MemTool 能够计算 hex 文件中数据的校验和。

- 1) 连接
- 2) Open File,
- 3) 点击 Device -> Calculate Device Checksum



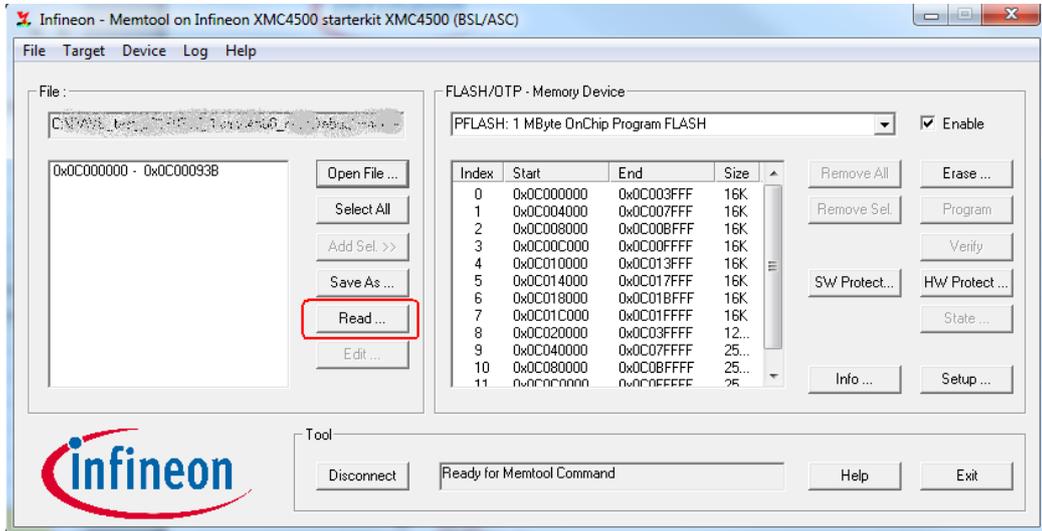
- 4) 计算完成后，能显示



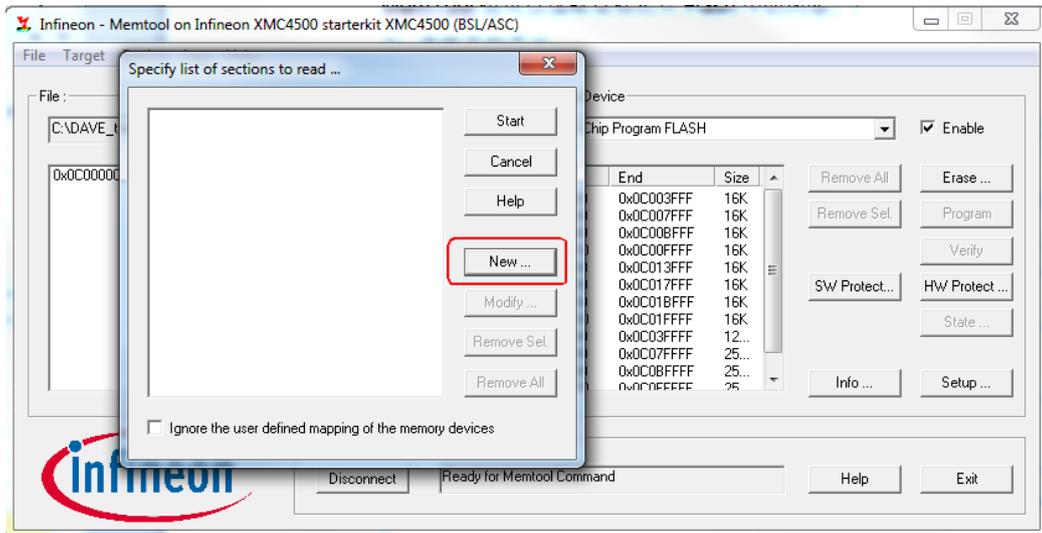
## 6.1.4 读取 Flash 内容

MemTool 也可以读取目标芯片 Flash 中的内容。

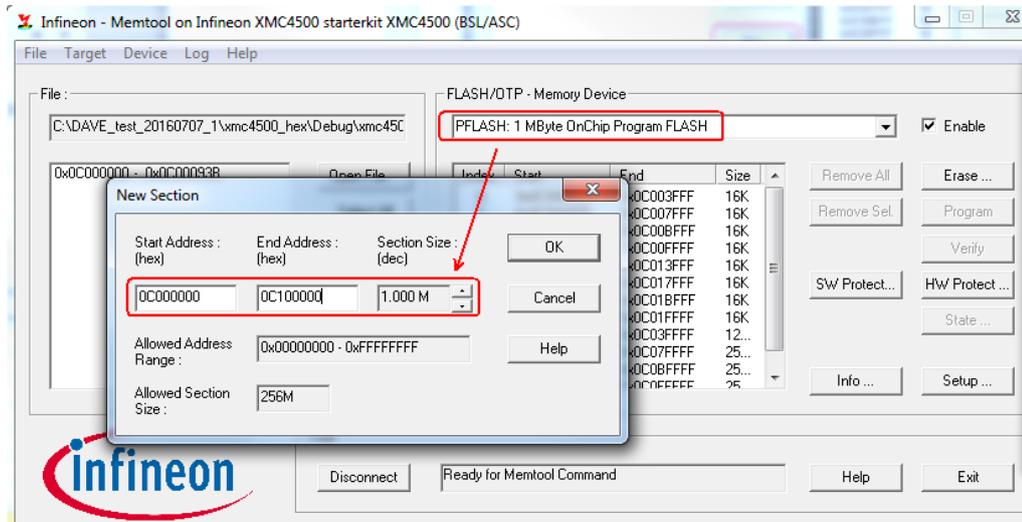
- 1) 连接目标芯片
- 2) 点击 Read



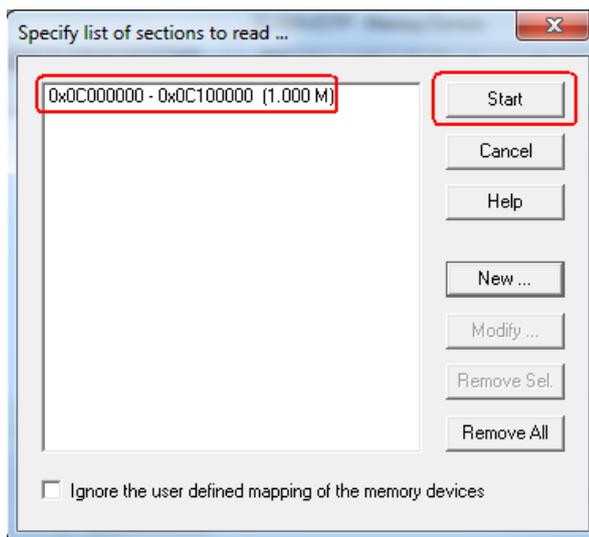
- 3) 点击 New



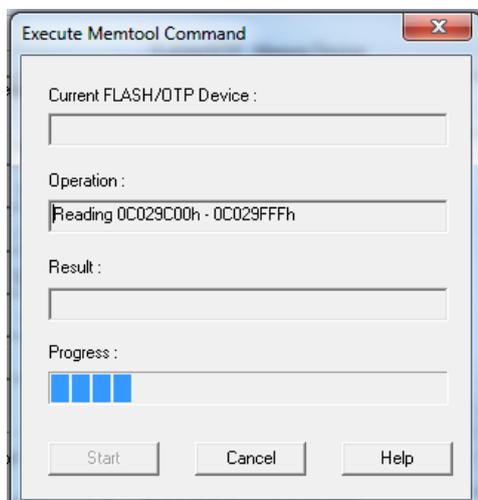
4) 根据 Flash 大小，填写数据首地址和尾地址后，点击 OK.



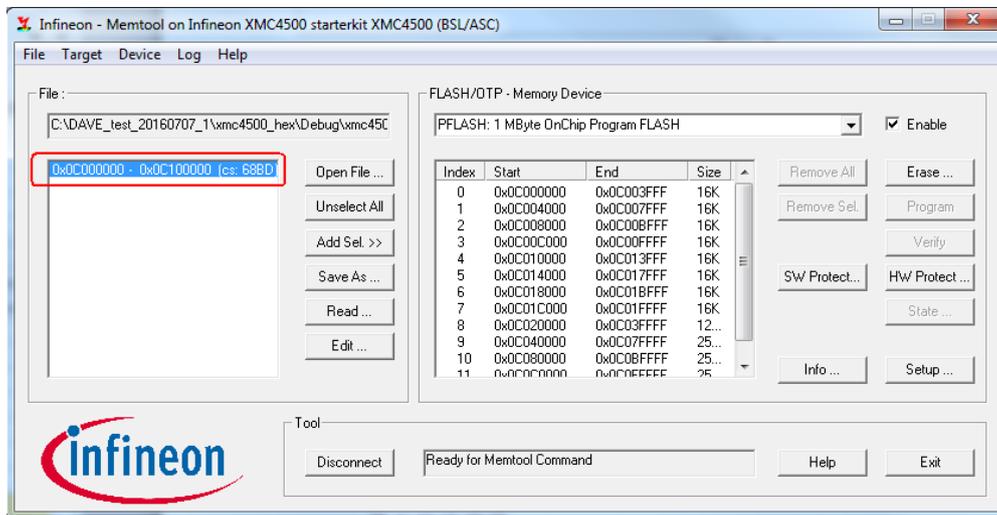
5) 此时会显示读取 Flash 的区域，点击 Start



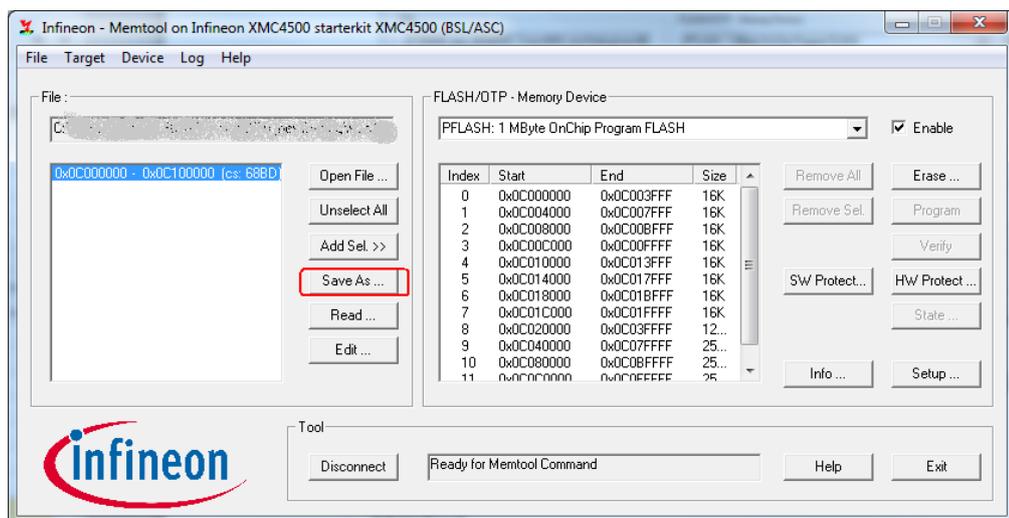
6) 开始读取文件



- 7) 文件读取完成后，点击 **Exit**，退出。这时候在下面窗口左侧显示读取 **Flash** 的地址和 **Flash** 中数据的校验和。



- 8) 点击 **Save As**，可以把读取的数据存成一个文件。这些数据可以继续去烧写别的 **MCU**。

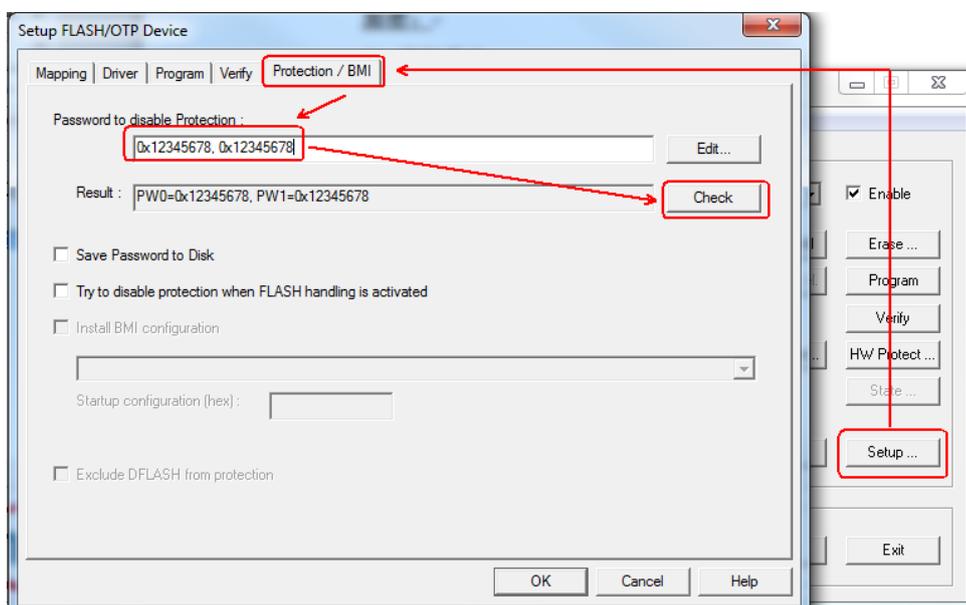


## 6.1.5 Flash 加密和解密

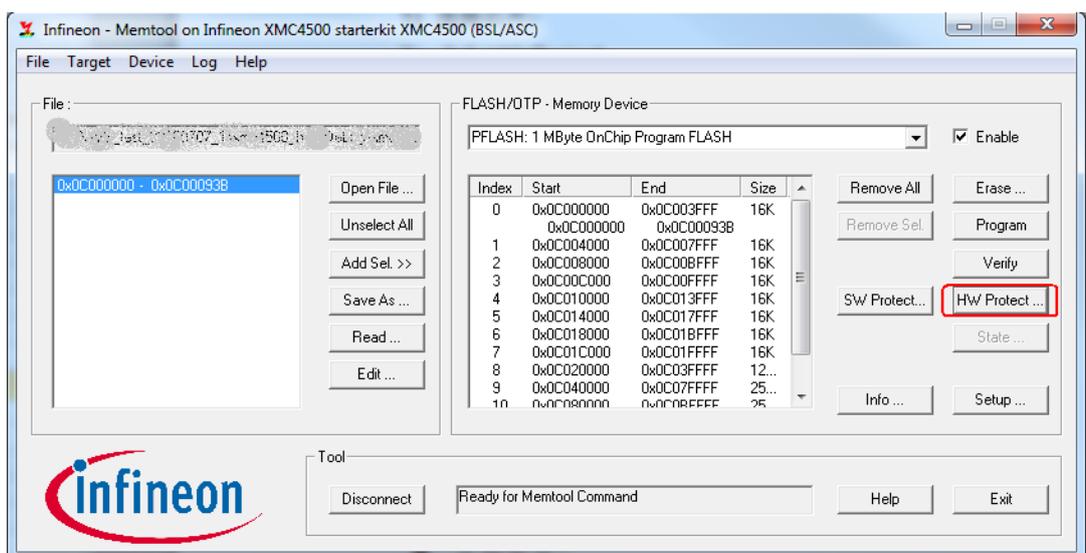
对于 XMC4000 系列的 MCU，可以通过 MemTool 对 Flash 进行加密和解密。

### 加密：

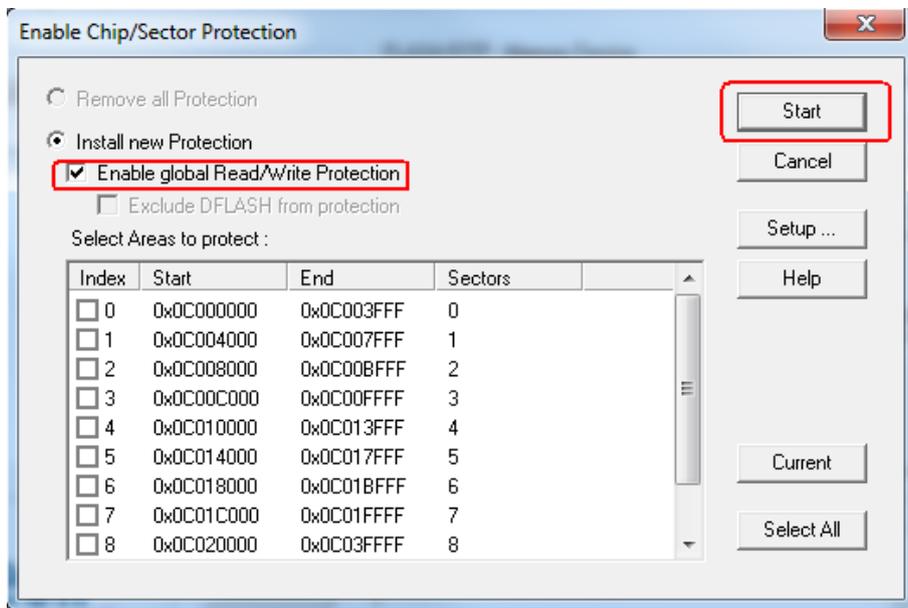
- 1) 连接芯片
- 2) 设置密码
  - a) 点击 Setup
  - b) 选择 Protection/BMI
  - c) 输入密码
  - d) 点击 Check



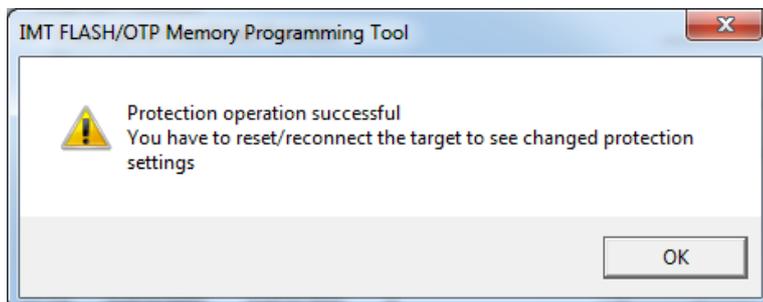
- 3) 点击 HW Project



4) 勾选“Enable Global Read/Write Protection”, 点击 Start

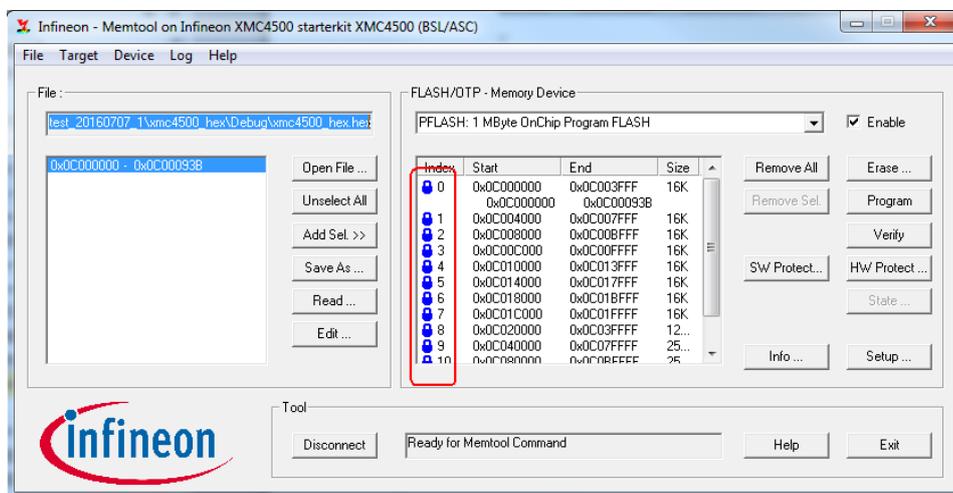


5) 在弹出来的对话框中, 点击 Yes, 开始加密, 加密完成后弹出下面窗口, 点击 OK



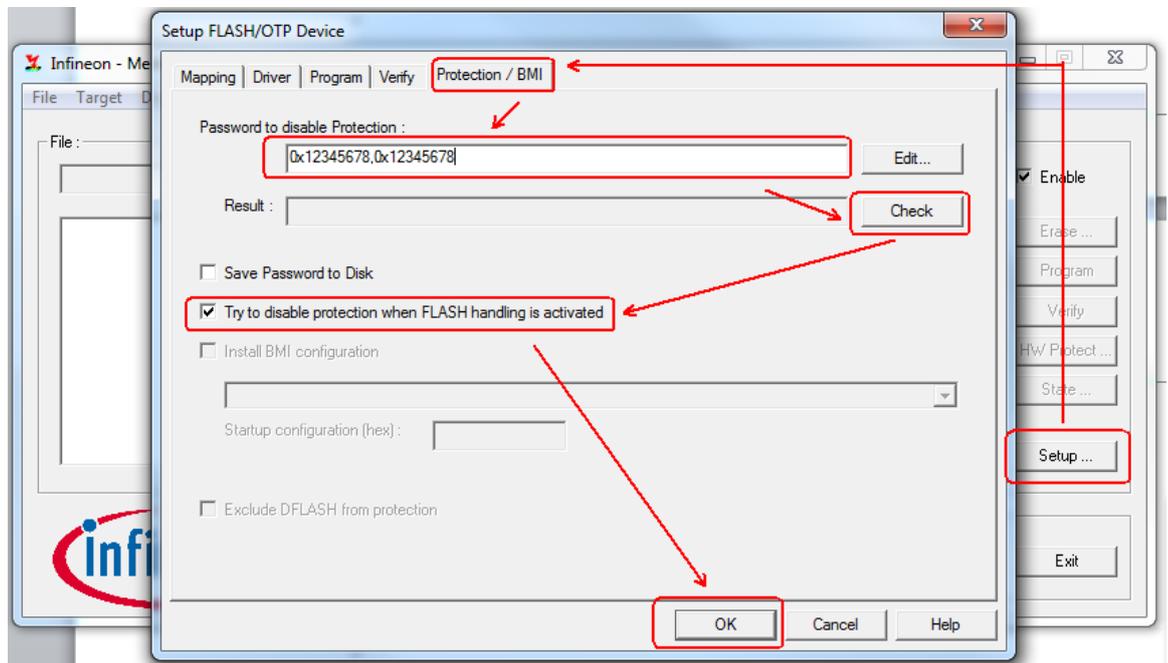
6) 复位 MCU, 或者先断电再上电

7) 重新连接 MCU, 发现各个 sector 已经加上了锁, 表示已经加密

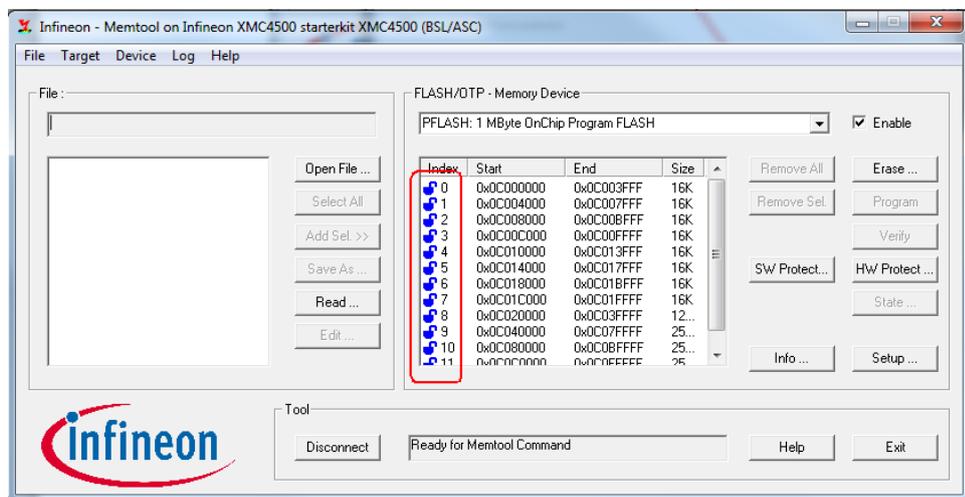


**暂时解密:**

- 1) 连接芯片
- 2) 输入密码
  - a) 点击 Setup
  - b) 选择 Protection/BMI
  - c) 输入密码
  - d) 点击 Check
  - e) 选择"Try to disable protection when FLASH handling is activated"
  - f) 点击 OK



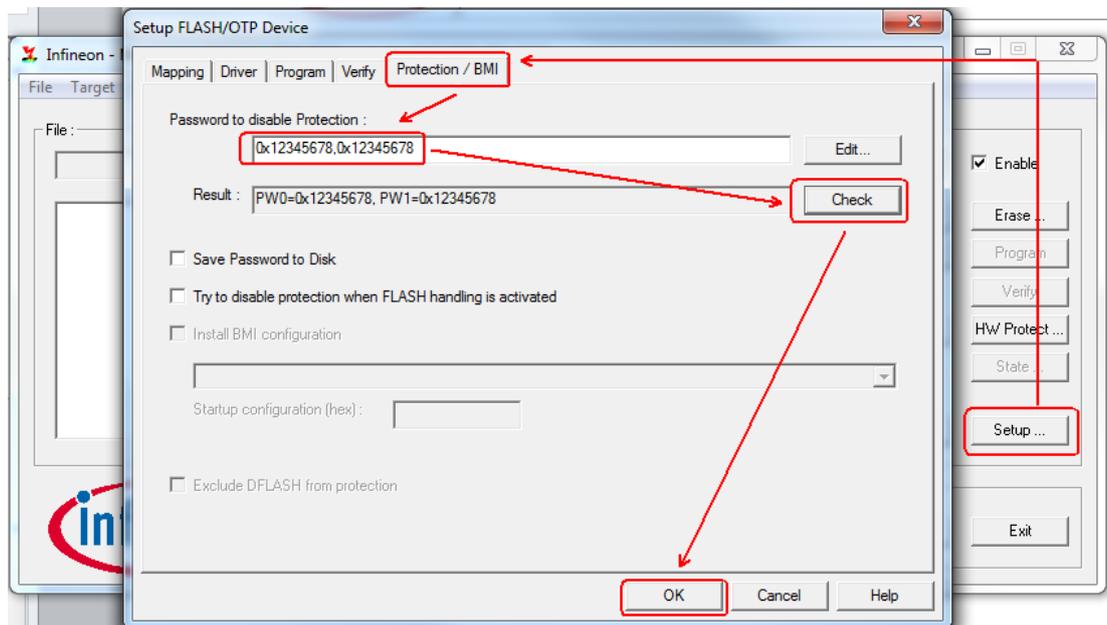
- 3) 这个时候可以看到 Flash 的锁已经打开，表示 Flash 可以操作了。



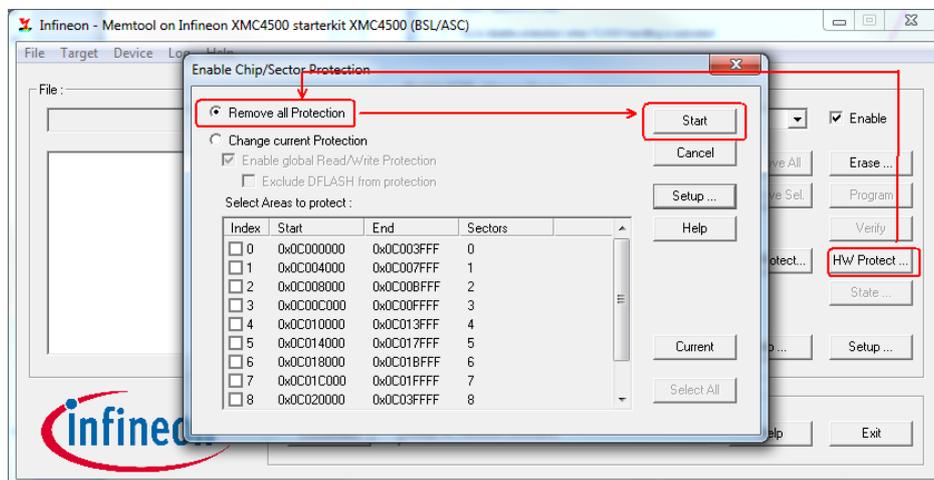
- 4) 接着就可以继续操作 Flash 了。

### 永久解密:

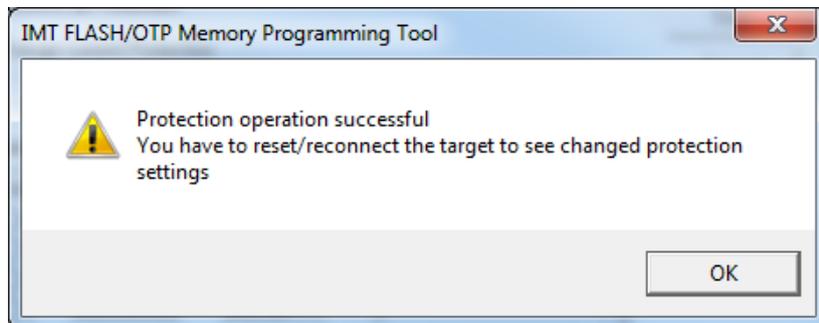
- 1) 连接芯片
- 2) 设置密码
  - a) 点击 Setup
  - b) 选择 Protection/BMI
  - c) 输入密码
  - d) 点击 Check
  - e) 点击 OK



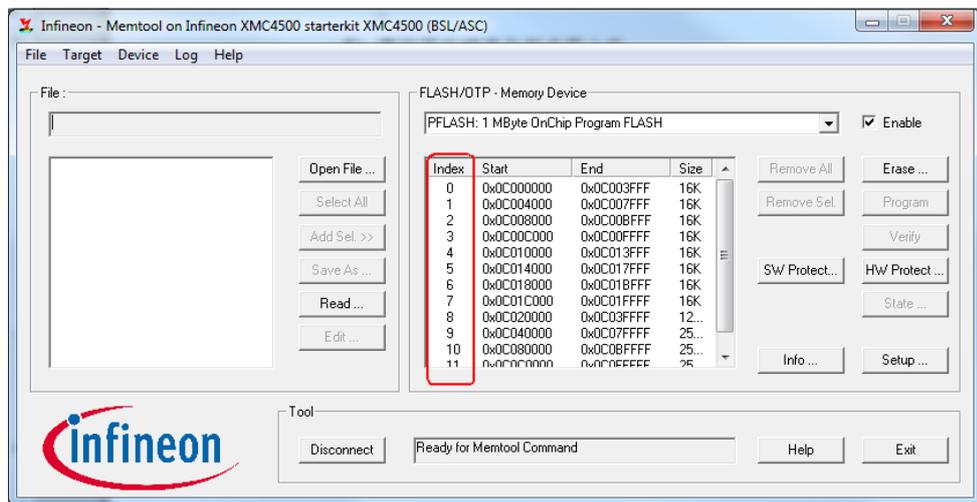
- 3) 设置密码
  - a) 点击 HW Protect
  - b) 选择 Remove all Protection
  - c) 点击 Start



- 4) 在弹出窗口，点击 Yes
- 5) 显示解除芯片保护成功



- 6) 复位芯片或者先断电再上电
- 7) 连接芯片，发现 Flash 的锁已经消失，表示 Flash 已经解除保护



## 6.2 XMC Flasher

XMC Flasher 是能通过 SWD 口烧写 XMC MCU 片内 Flash 的工具。XMC Flasher 下载地址如下：

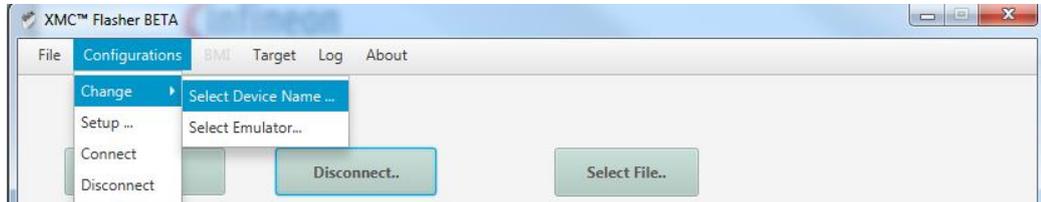
[www.infineon.com/xmcflasher](http://www.infineon.com/xmcflasher)

在使用 XMC Flasher 之前，首先需要安装下面两个软件：

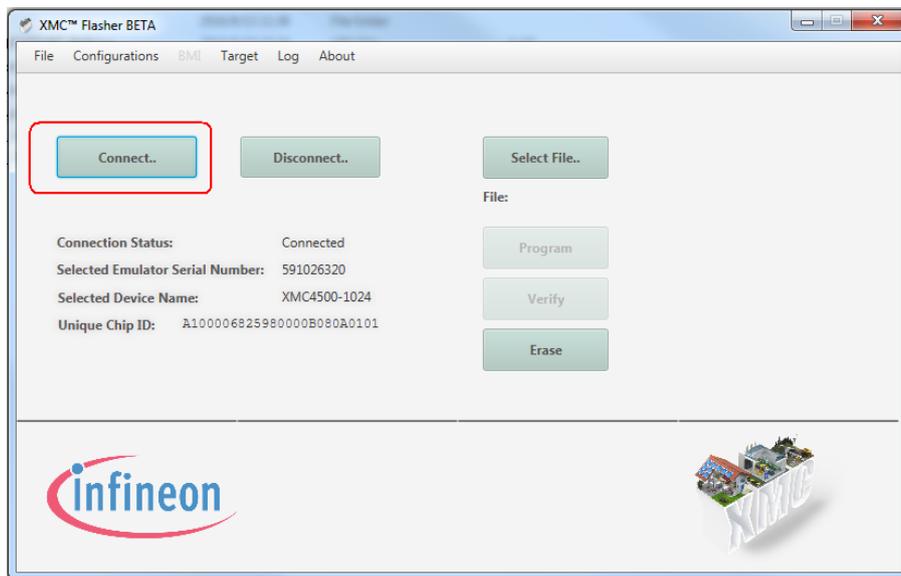
- 1) Oracle JAVA™ JRE 1.8.0\_72 or higher  
<http://www.oracle.com/technetwork/java/javase/downloads/index.html>
- 2) Segger JLINK software 5.12 or higher  
<https://www.segger.com/jlink-software.html>

## 6.2.1 烧写流程

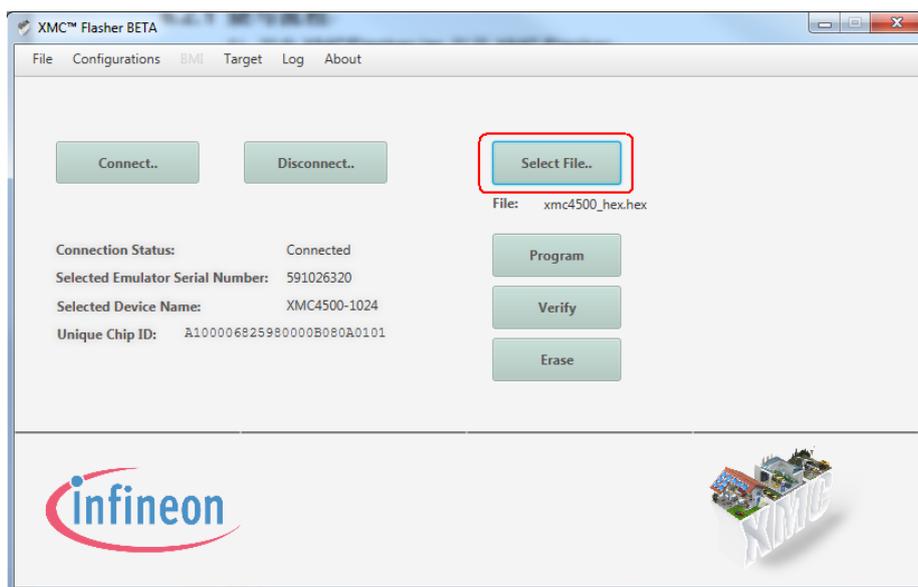
- 1) 双击 XMCFlasher.jar, 打开 XMC Flasher
- 2) 选择设备



- 3) 连接芯片，连接成功后会显示设备名字和设备唯一 ID



- 4) 选择 Hex 文件，选择后会在下面显示该 Hex 文件名字

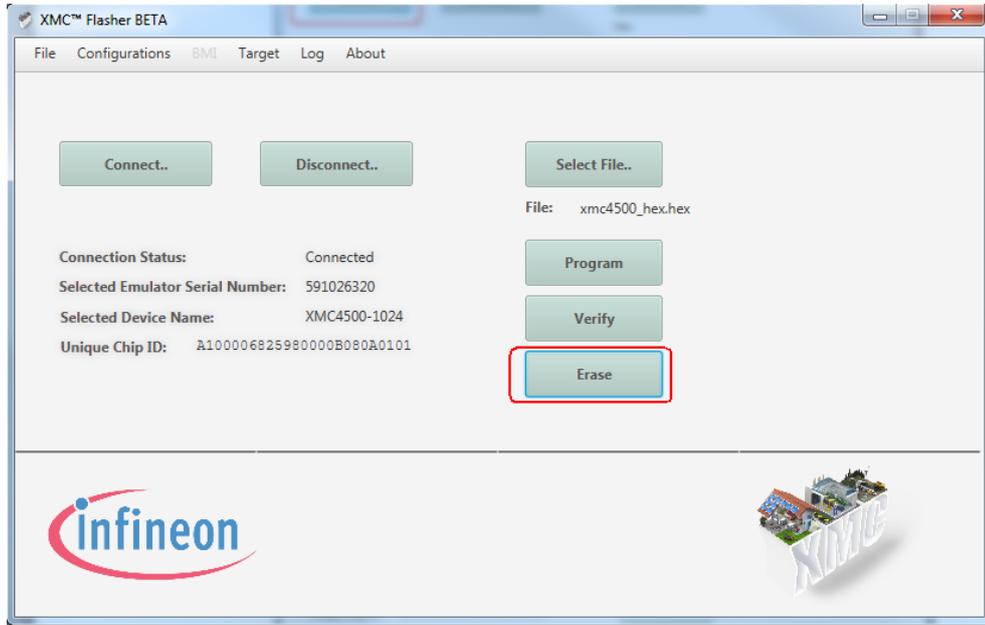


- 5) 点击 Program，启动烧写。

## 6.2.2 擦除芯片

这里的擦除芯片指的是整片 Flash 的擦除(chip erase)。

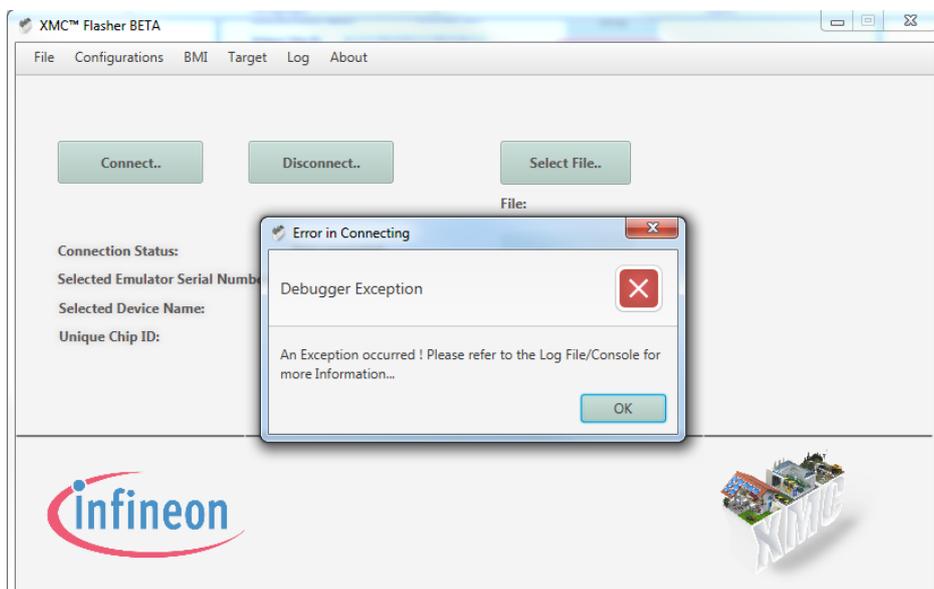
- 1) 连接芯片
- 2) 点击 Erase



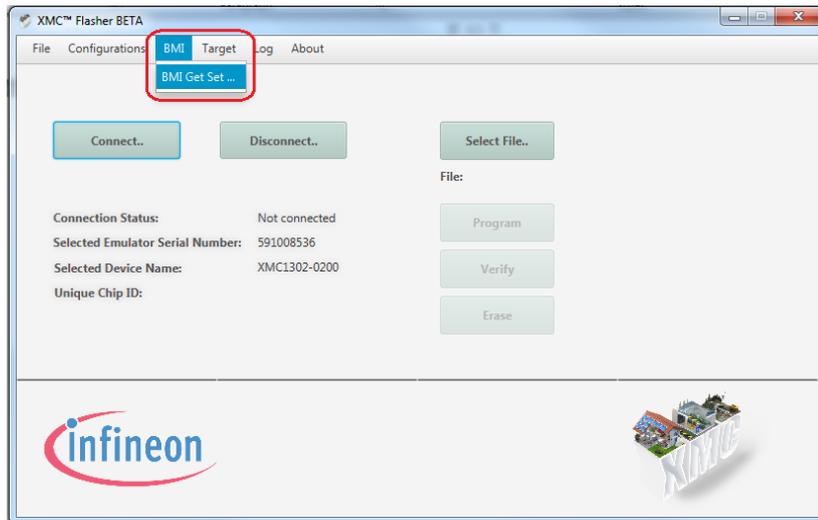
## 6.2.3 修改 BMI

XMC Flasher 还具有修改 BMI 的功能。XMC1000 系列的产品出厂时的 BMI 模式是 ASC\_BSL 模式，在这个模式下，不能调试芯片，需要把 MCU 的模式修改为 SWD 或者 SPD 后，才能进行调试。

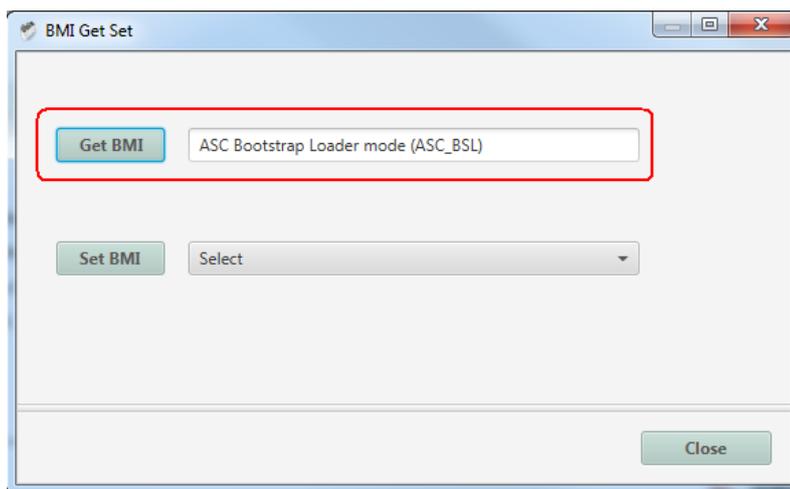
- 1) 首次连接芯片
- 2) 如果芯片的 BMI 模式是 ASC\_BSL 模式，则会报错。



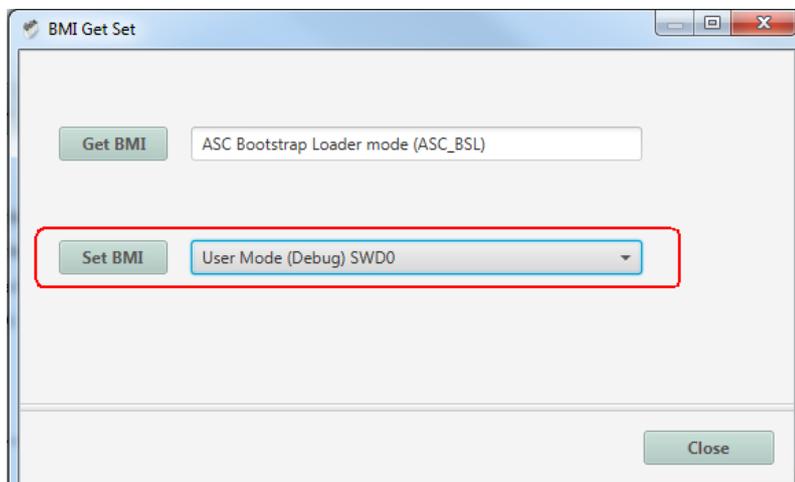
3) 点击 OK, 点击 BMI->BMI Get Set...



4) 点击 Get BMI 可以获取芯片目前的 BMI 模式



5) 点击 Set BMI 可以设置 BMI 模式，例如把模式设为 USER\_MODE (SWD0)



## 7 uC/Probe

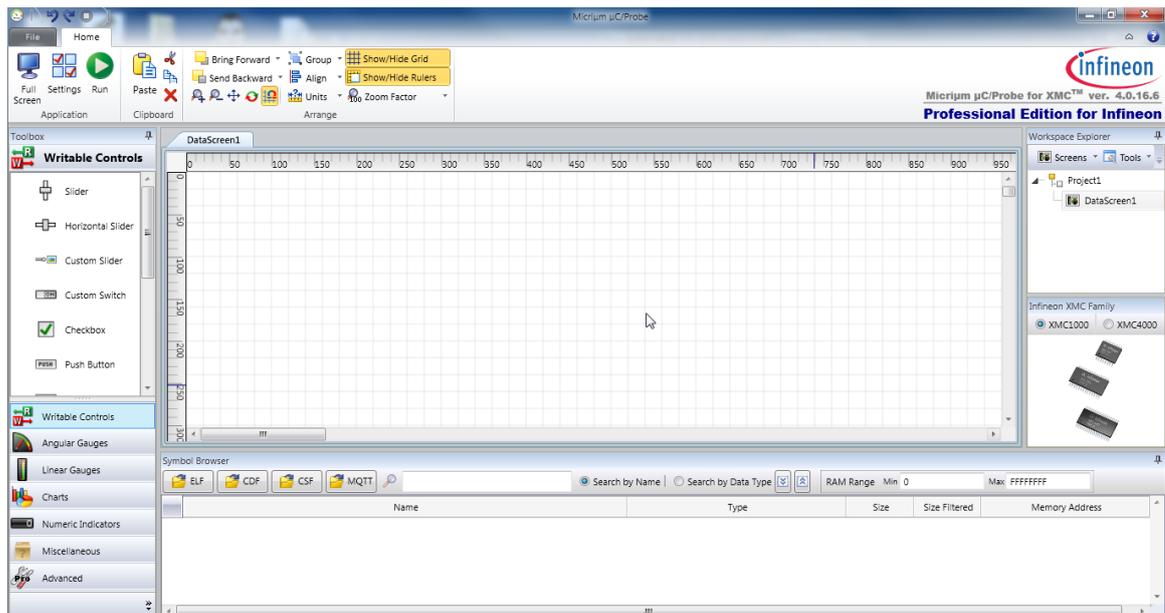
### 7.1 uC/Probe 简介

英飞凌最近发布了一款新的调试工具 – uC/Probe，可以实时的观察变量或者外设寄存器的变化。它以各种控件的形式表示实际的量值，例如文本框，仪表盘，甚至示波器。这种形式的调试会对一些需要动态观测参数的应用带来很大的方便，例如电机和数字电源应用。但是使用它之前，需要首先 download 程序到 MCU，然后 load elf 文件到 uC/Probe。具体请看附件使用说明。

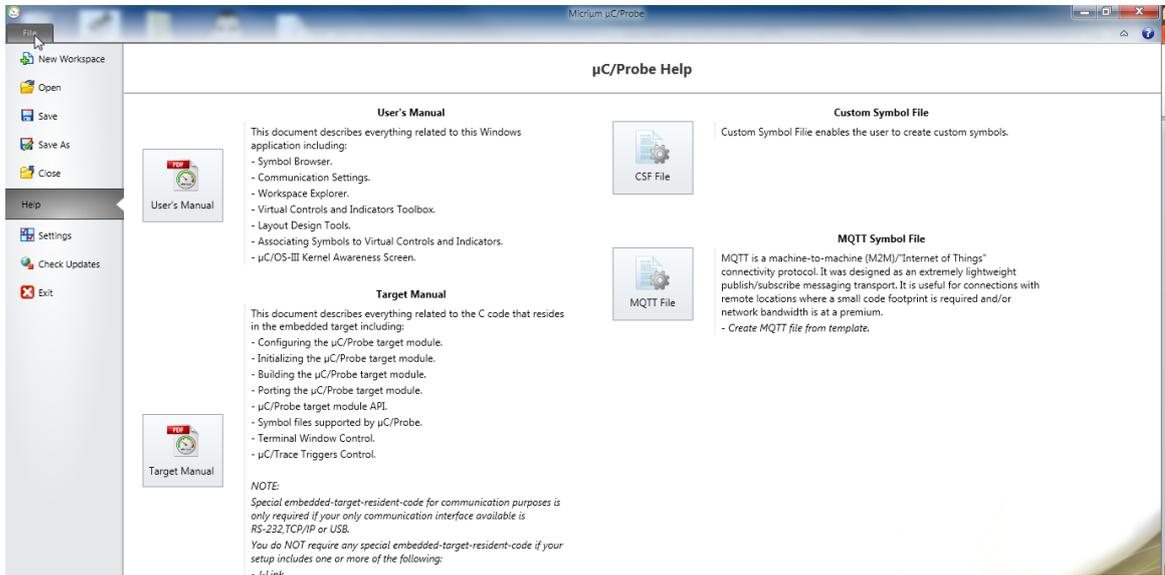
uC/Probe 这个工具是 Micrium 开发的，一般来说，开发者需要使用需向 Micrium 购买 License. 我们提供的版本是 Micrium 给 Infineon 定制的，即只能调试 XMC MCU，并且是免费提供给 XMC 用户使用的。

### 7.2 uC/Probe 使用说明

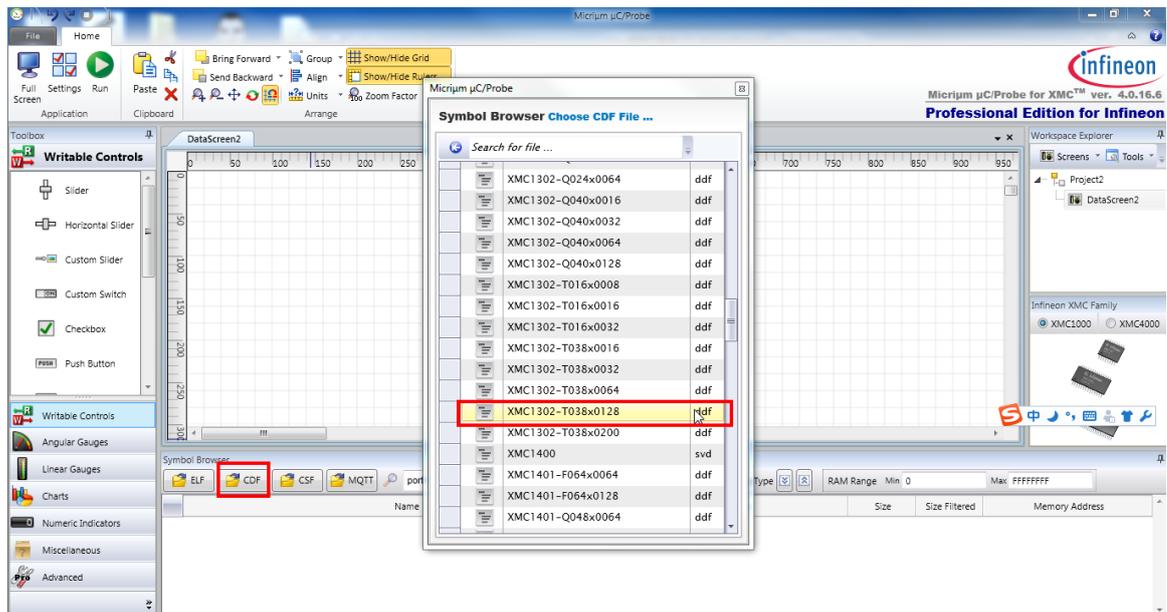
#### 1) 打开 uC/Probe



2) 鼠标点击左上角'File'，可以看到两个 pdf 手册描述细节，再点一下 File 返回



3) 点击 CDF，在列表里选择对应 MCU 型号



4) 以 Keil 例程为例

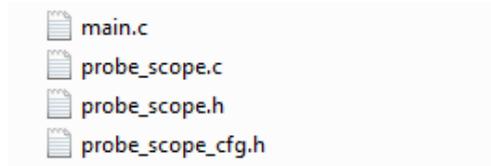
a) 生成一个 KEIL 的 XMC1300 的 Blinky 工程，做测试用途



b) 解压缩安装文件里附带的 Micrium-Probe-TargetCode.Zip,找到下面路径

- Micrium-Probe-TargetCode\Micrium\Software\uC-Probe\Target\Scope

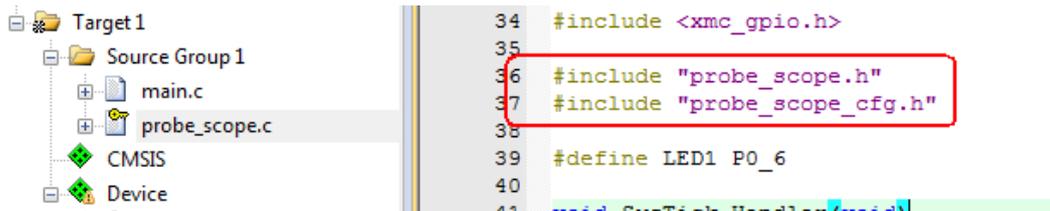
c) Copy scope 下三个文件到测试工程目录，添加 C 文件到 Keil workspace



d) 打开 option, 添加路径

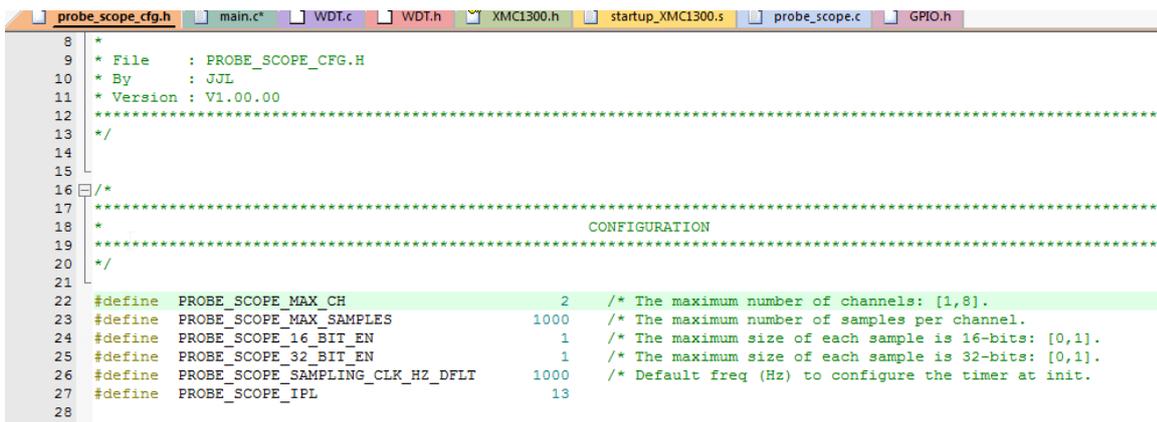
- .\

e) Include 两个头文件



5) 配置 cfg 文件

a) 打开 cfg 头文件，配置示波器通道，根据自己芯片的内存大小选择通道数和采样数，这里把默认的 8 通道改成 2 通道。



## 6) 添加程序

- A. 在 Main.c 里面调用初始化函数 ProbeScope\_Init(10000).这里的形参 10000 代表下面的采样程序每 0.1mS (10000HZ) 就会去采样需要观察的变量。
- B. 修改 SysTick\_Config()参数, 每 0.1ms 产生一次 System Tick 的中断
- C. 在 System Tick 中断服务函数中添加 ProbeScope\_Sampling()函数
- D. 变量 Count 和 Count1 用来测试

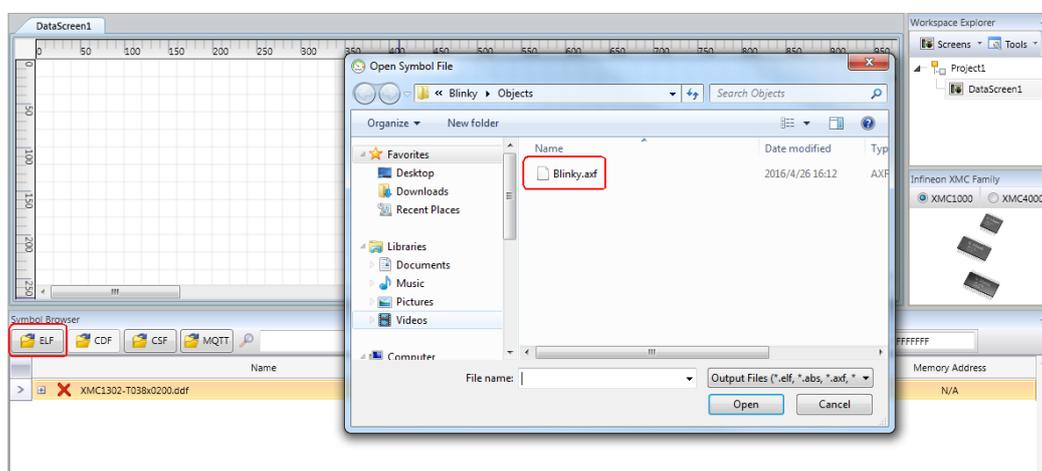
```

39 #define LED1 P0_6
40
41 uint16_t Count = 0, Count1 = 0; D
42
43 void SysTick_Handler(void)
44 {
45     Count++;
46     Count1 += 2; D
47     XMC_GPIO_ToggleOutput(LED1);
48     ProbeScope_Sampling(); C
49 }
50
51
52 int main(void)
53 {
54     ProbeScope_Init(10000); A
55
56     XMC_GPIO_SetMode(LED1, XMC_GPIO_MODE_OUTPUT_PUSH_PULL);
57
58     /* System timer configuration */
59     SysTick_Config(SystemCoreClock/10000); B
60
61     while(1)
62     {
63         /* Infinite loop */
64     }
65 }
66

```

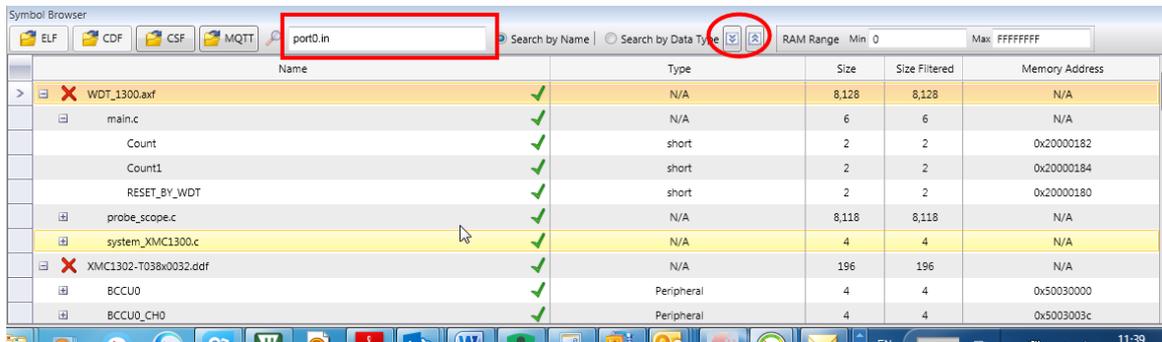
## 7) 载入对应的编译器仿真文件

- a) 编译完成后, download 到 XMC1300BOOTKIT, 切换回 uC\_Probe
- b) Keil 用户选择 axf 文件打开 (DAVE 用户选择 elf, 其他找对应的文件后缀)



## 8) Symbol Browser 窗口

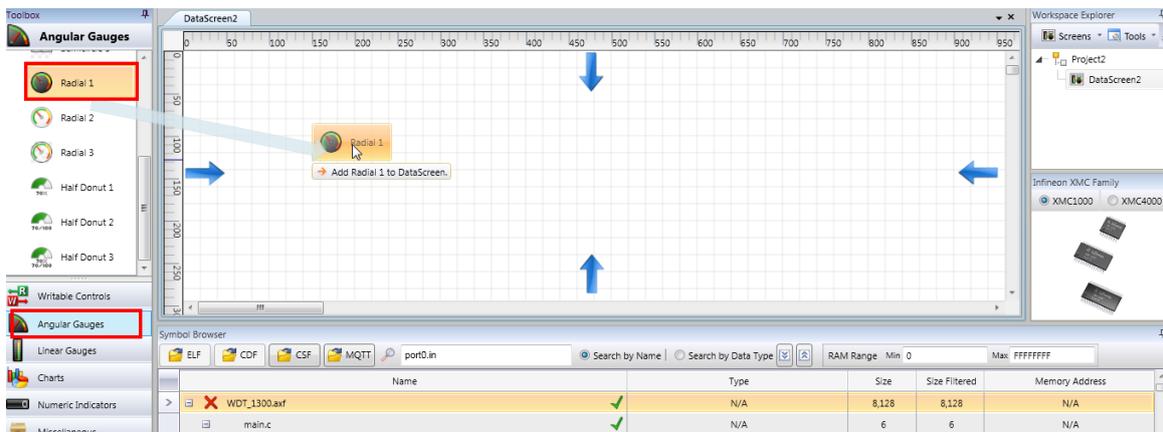
- a) 在 Symbol Browser 窗口可以看到相应的变量和外设
- b) 查找窗口输入名字可以快速找到寻找的内容
- c) 圆圈处可以关闭和展开分支



## 仪表盘功能

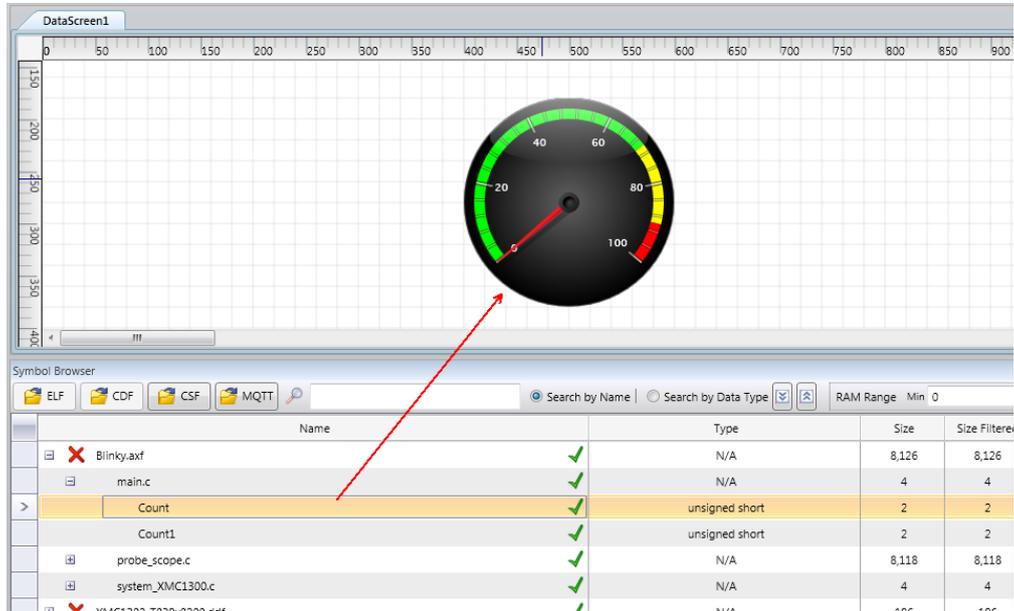
### 1) 增加一个仪表显示

- a) 选择 Angular Gauges, 选择一个仪表, 拖到 datascreen 里面



## 2) 关联变量和仪表

a) 在变量里选择一个拖入仪表界面

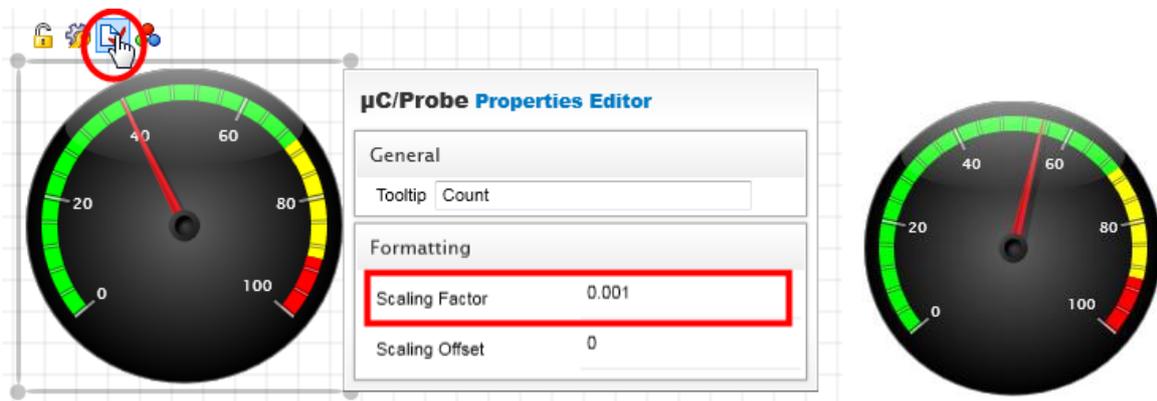
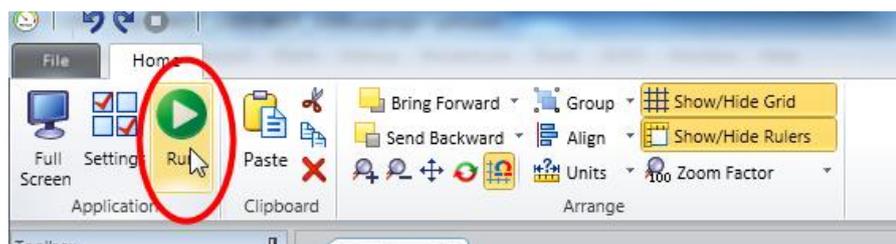


## 3) 设定显示比例

a) 确认 Bootkit 里面程序已经运行

b) 由于 count 设为 uint16\_t, 从 0-65535 循环, 因此把比例 Scaling Factor 设为 0.001, 这样指针转到 65 到 66 之间就回到 0

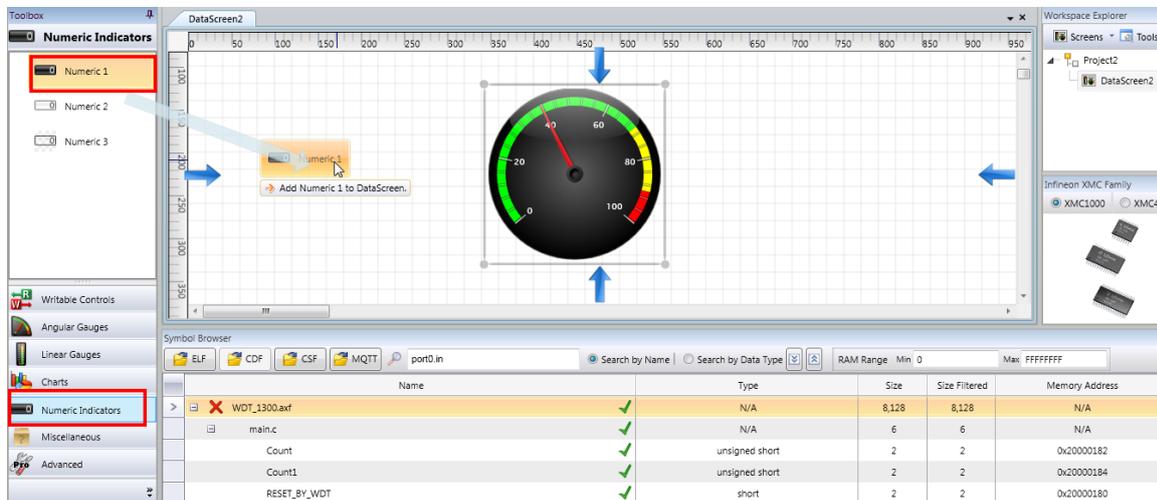
c) 设定完成后, 点击 Run, 可以看到指针动态从 0 到 65 移动



## 数字表功能:

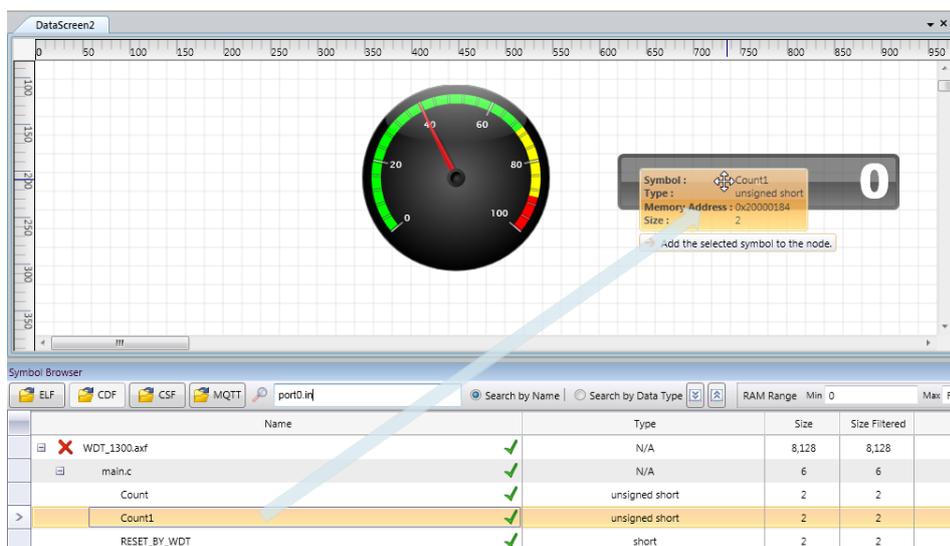
1) 增加一个数字表显示

a) 选择 Numeric Indicators, 鼠标拖一个数字表到 DataScreen



2) 关联变量到数字表

a) 拖变量 Count1 到数字表



3) 显示结果

a) 点击 Run 后, 可以看到变量 Count1 显示到数字表, 不断变化

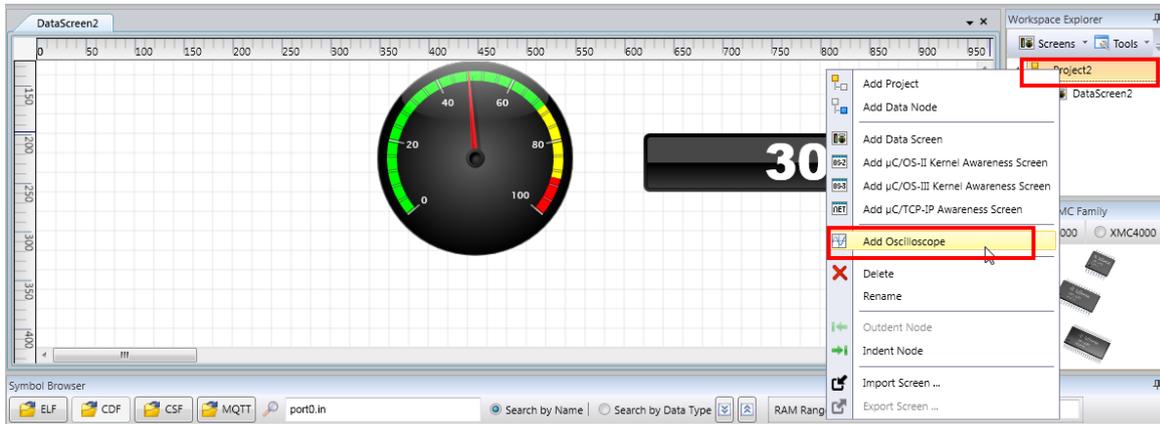


**20980**

## 示波器功能：

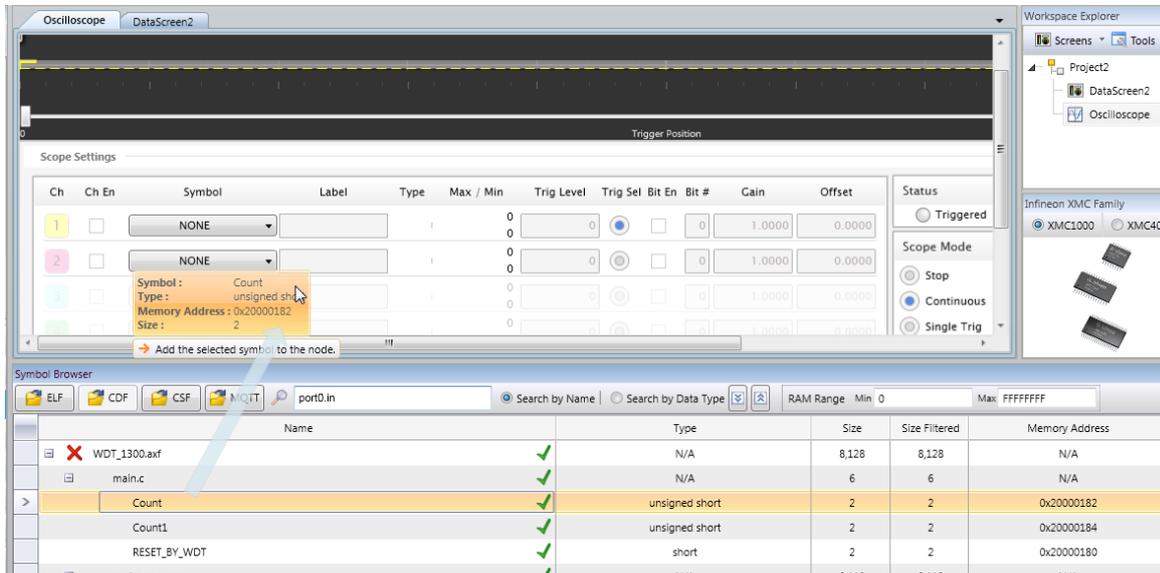
### 1) 增加示波器功能

a) 右键点击 project, 选择 Add Oscilloscope



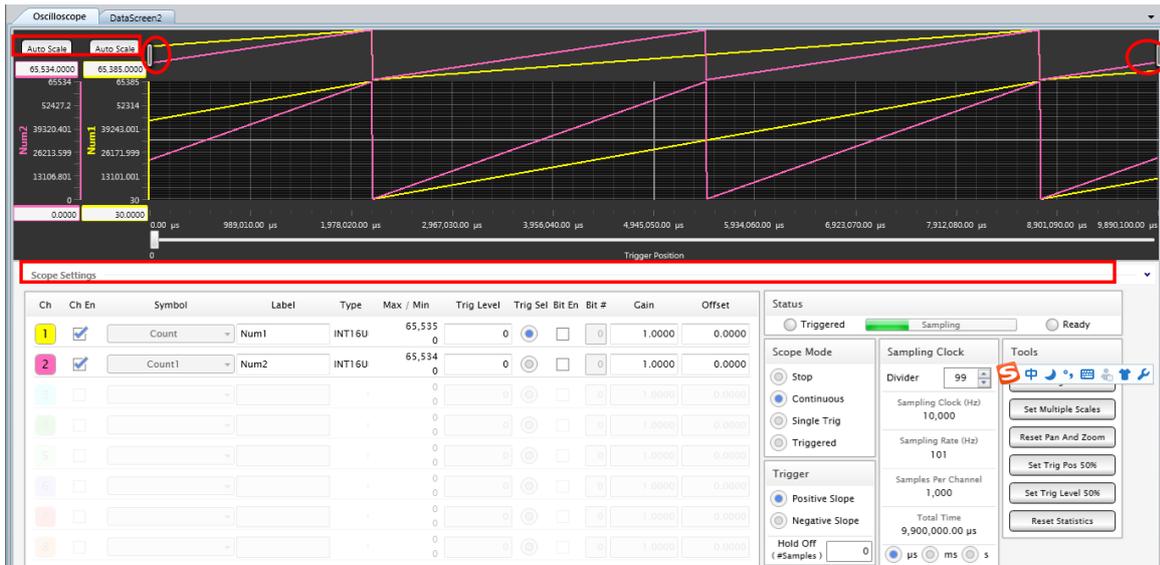
### 2) 把变量添加到示波器

a) 用鼠标把变量 count, count1 分别拖到 Scope Setting

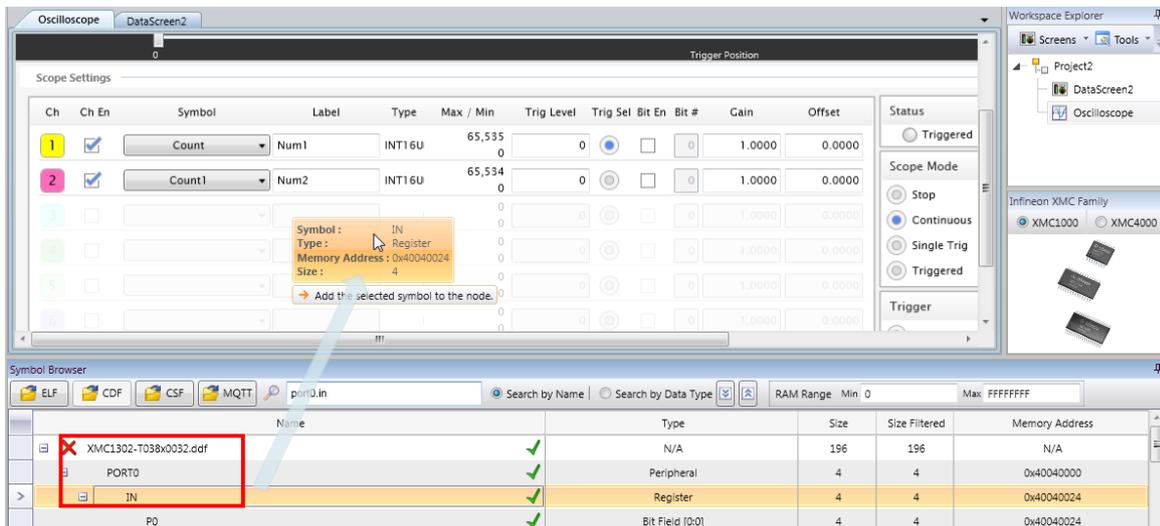


### 3) 幅度和时间轴调节

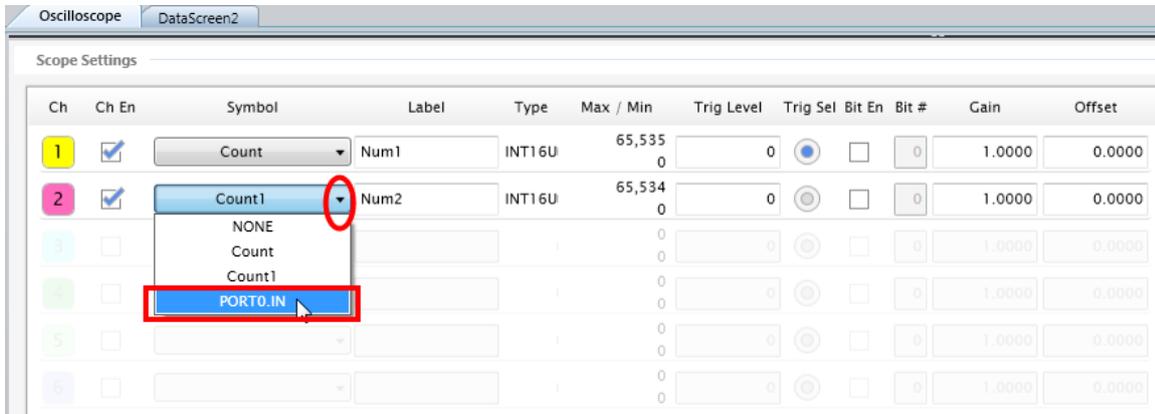
- 点击 **Auto\_Scale** 把幅度调整到合适范围
- 时间轴在 **Sampling Clock divide** 调整
- 触发和停止等见操作界面，和正常示波器类似
- 双击 **scope Setting** 可以放大示波器到全屏
- 拖拽上方圆圈处可以进行局部放大



### 4) 拖拽寄存器 PORT0.IN 到示波器设置



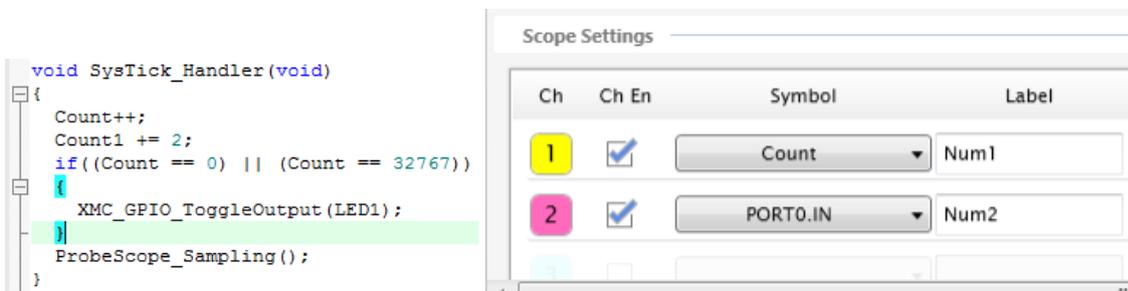
5) 下拉菜单选择 PORT0.IN



6) 选择 Bit, 第 6 位



7) 输出结果

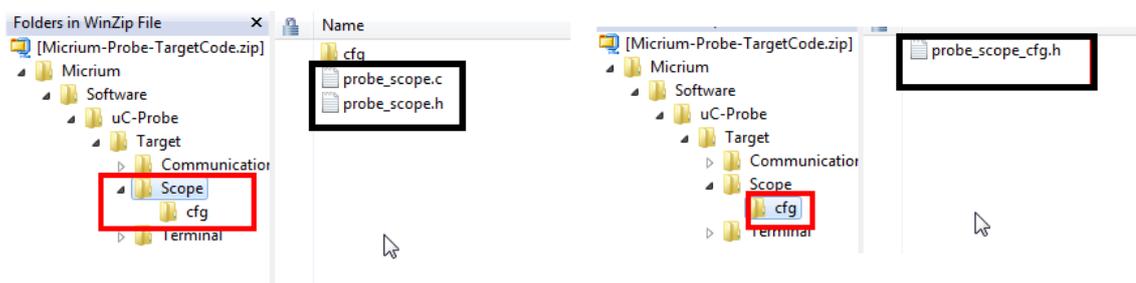


### 7.3 uC/Probe 文件资源

安装程序包:

File Name	Date	Type	Size	Description
Micrium-Probe-TargetCode	2016/4/5 22:00	WinZip File	439 KB	示波器辅助功能代码
Micrium-uC-Probe-Setup-Release-4.0.16.6_INFINEON	2016/4/10 18:39	Application	105,793 KB	安装程序
Release_Update.	2016/2/22 6:43	Text Document	3 KB	
xmc4700_relaxkit_waveform_generator	2016/4/5 22:00	WinZip File	4,479 KB	例程

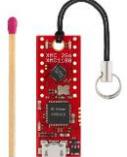
示波器辅助功能代码位置:

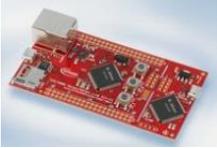


## 8 开发套件

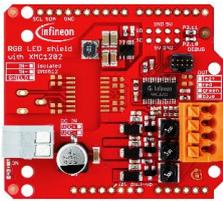
英飞凌提供 3 类开发套件：入门套件，应用套件和演示套件。可以浏览下面网站，查看套件具体情况。

[www.infineon.com/xmc-dev](http://www.infineon.com/xmc-dev)

套件类型	型号	特性	图片
XMC1000 入门套件	KIT_XMC_2GO_XMC1100_V1	<ul style="list-style-type: none"> <li>- XMC1100-Q024</li> <li>- 板载J-Link</li> <li>- 所有引脚均引出来</li> </ul>	
	KIT_XMC11_BOOT_001	<ul style="list-style-type: none"> <li>- XMC1100-T038</li> <li>- 板载J-Link</li> <li>- 所有引脚均引出来</li> <li>- Arduino接口</li> </ul>	
	KIT_XMC12_BOOT_001	<ul style="list-style-type: none"> <li>- XMC1200-T038</li> <li>- 板载J-Link</li> <li>- 所有引脚均引出来</li> <li>- 8 x LED</li> <li>- 1 x Pentimeter</li> <li>- 扩展接口</li> </ul>	
	KIT_XMC13_BOOT_001	<ul style="list-style-type: none"> <li>- XMC1300-T038</li> <li>- 板载J-Link</li> <li>- 所有引脚均引出来</li> <li>- 8 x LED</li> <li>- 1 x Pentimeter</li> <li>- 扩展接口</li> </ul>	
	KIT_XMC14_BOOT_001	<ul style="list-style-type: none"> <li>- XMC1400-Q064</li> <li>- 板载J-Link</li> <li>- 所有引脚均引出来</li> <li>- 8 x LED</li> <li>- 1 x Pentimeter</li> <li>- 扩展接口</li> <li>- 1 x CAN</li> </ul>	
XMC4000 入门套件	KIT_XMC45_RELAX_LITE_V1	<ul style="list-style-type: none"> <li>- XMC4500-F144</li> <li>- 板载J-Link</li> <li>- 4 x SPI-Master, 3x I2C, 3 x I2S,</li> <li>- 3 x UART, 2 x CAN, 17 x ADC</li> <li>- 2 x user button and 2 x user LED</li> </ul>	

	KIT_XMC45_RELAX_V1	<ul style="list-style-type: none"> <li>- XMC4500-F144</li> <li>- 板载J-Link</li> <li>- 4 x SPI-Master, 3x I2C, 3 x I2S, 3 x UART, 2 x CAN, 17 x ADC</li> <li>- Ethernet PHY and RJ45 jack</li> <li>- 32 Mbit Quad-SPI Flash</li> <li>- microSD card slot</li> </ul>	
	KIT_XMC47_RELAX_LITE_V1	<ul style="list-style-type: none"> <li>- XMC4700-F144</li> <li>- 板载J-Link</li> <li>- 2 x user button and 2 x user LED</li> <li>- 3.3V Arduino接口</li> </ul>	
	KIT_XMC47_RELAX_V1	<ul style="list-style-type: none"> <li>- XMC4700-F144</li> <li>- 板载J-Link</li> <li>- Quad SPI Flash</li> <li>- microSD Card Slot</li> <li>- Ethernet PHY and RJ45 Jack</li> <li>- CAN node</li> </ul>	
六角板 应用套件	KIT_XMC45_EE1_002 KIT_XMC45_EE2_001 KIT_XMC44_EE1_001 KIT_XMC42_EE1_001	<ul style="list-style-type: none"> <li>- XMC4500/XMC4400/XMC4200</li> <li>- 板载J-Link</li> <li>- COM接口</li> <li>- ACT接口</li> <li>- HMI接口</li> </ul>	
	KIT_XMC4X_HMI_OLED_001	<ul style="list-style-type: none"> <li>- HMI接口</li> <li>- Micro SD Card Slot</li> <li>- Matrix OLED Display</li> <li>- 2.5 mm stereo receptacle</li> <li>- Stereo audio codec</li> </ul>	
	KIT_XMC4X_COM_ETH_001	<ul style="list-style-type: none"> <li>- COM接口</li> <li>- 10/100Mbps Ethernet (RJ45)</li> <li>- CAN transceiver with CAN connector</li> <li>- RS-485 transceiver, full-duplex, bootable</li> <li>- I2C based IO expander</li> </ul>	
	KIT_XMC4X_MOT_GPDLV_001	<ul style="list-style-type: none"> <li>- ACT接口</li> <li>- 3 phase low voltage half-bridge inverter</li> <li>- Gate Driver IC (6ED003L02-F2)</li> <li>- Delta-sigma modulator</li> <li>- Inductive resolver interface</li> <li>- Hall sensor interface</li> </ul>	
	KIT_XMC4X_AUT_ISO_001	<ul style="list-style-type: none"> <li>- ACT接口</li> <li>- ISOFACE OUT, up to 8 channels</li> <li>- ISOFACE IN, up to 8 channels</li> <li>- I2C based IO expander up to 8 channels</li> </ul>	

电机 应用套件	KIT_XMC1X_AK_MOTOR_001	主控板 - XMC1300 - 板载J-Link  电机板 - 12 – 24V, up to 3A - On board 3-phase motor (24V, 15W) with hall sensors, Optional encoder interface - Power supply 24V, 1A	
	XMC_LVPB V1.0	控制板: - XMC1300 - POSIF电机位置接口 - RS232通讯接口 - SPI通讯接口支持片外DAC扩展 - 调试接口 8针J-LINK接口 - 20pin功率板接口  功率板: - 功率电源接口 - 电机输出接口 - 辅助电源 - 20pin主控板接口	
LED 应用套件	KIT_XMC1X_AK_LED_001	控制板: - XMC1200 - 板载J-Link  Colour LED Card: - 3 RGB LEDs, 10mA - Connectivity: DALI, DMX, RF - Ambient light sensor  White LED Card: - 20 LEDs on 4 strings, 20mA - Connectivity: DALI, RF - Ambient light sensor - Temperature sensor	
	KIT_XMC1_LED_CC_EXP_001	- Low cost, flicker-free, smart and connected single channel LED driver demonstrator - 15W Single Channel CCM DC/DC Buck converter in peak-current control mode - Software-based automatic adaptation to different input DC voltages and LED forward voltages - Isolated DALI Interface card. DALI Software Stack available	
EtherCAT 应用套件	KIT_XMC48_RELAX_ECAT_V1	- XMC4800-F144 - 板载J-Link - Quad SPI Flash - microSD Card Slot - Ethernet PHY and RJ45 Jack - CAN node - EtherCAT® node	

Arduino 应用套件	KIT_LED_XMC1202_AS_01	<ul style="list-style-type: none"> <li>- Compatible with Arduino Uno R3 and XMC1200 Boot Kit</li> <li>- Easy configurable for various light engines and any input voltage (within operating conditions)</li> <li>- Wide DC input voltage range</li> <li>- Simple I 2C interface</li> </ul>	
	DC- MOTORCONTR_BTN8982	<ul style="list-style-type: none"> <li>- Compatible with Arduino Uno R3</li> <li>- Capable of high frequency PWM</li> <li>- Adjustable slew rates for optimized EMI by changing external resistor</li> <li>- Driver circuit with logic level inputs</li> <li>- Diagnosis with current sense</li> </ul>	
数字电源 应用套件	KIT_XMC_DP_EXP_01	主控板： <ul style="list-style-type: none"> <li>- XMC1300/XMC4200</li> <li>- 板载 J-Link</li> </ul> 数字电源开发板： <ul style="list-style-type: none"> <li>- Synchronous buck converter</li> <li>- On-board switches for step load testing</li> <li>- PMBus communication</li> <li>- Network analyzer ready for stability analysis</li> <li>- Option to connect a second buck for interleaving control</li> </ul>	
750W 电机控制演 示套件	KIT_XMC750WATT_MC_AK_V1	主控板： <ul style="list-style-type: none"> <li>- XMC1300/XMC4400</li> <li>- 板载J-Link</li> </ul> 功率板： <ul style="list-style-type: none"> <li>- 110-230 V / 750 W</li> <li>- Input Filter and Active PFC</li> <li>- 3-Phase Inverter by Reverse Conduction IGBT</li> </ul>	
600W LLC 演示套件	EVAL_600W_12V_LLC_C7_d	<ul style="list-style-type: none"> <li>- Output voltage: 12V</li> <li>- Output current: 50A</li> <li>- Peak efficiency @ 50% load &gt; 97,4%</li> <li>- Efficiency @ 10% load &gt; 95%</li> </ul>	
3KW Dual LLC 演 示 套件	EVAL_3KW_2LLC_C7_20	<ul style="list-style-type: none"> <li>- Peak efficiency &gt;98.4%</li> <li>- Flat efficiency plot from 10% to 100% load</li> <li>- Flexible design adjustment and fine tuning through the GUI</li> </ul>	

## 9 XMC 相关链接

XMC 主页	<a href="http://www.infineon.com/xmc">www.infineon.com/xmc</a>
XMC4000 主页	<a href="http://www.infineon.com/xmc4000">www.infineon.com/xmc4000</a>
XMC1000 主页	<a href="http://www.infineon.com/xmc1000">www.infineon.com/xmc1000</a>
XMC 开发套件	<a href="http://www.infineon.com/xmc-dev">www.infineon.com/xmc-dev</a>
XMC 生态系统	<a href="http://www.infineon.com/xmc/ecosystem">www.infineon.com/xmc/ecosystem</a>
MatrikonOPC UA Embedded SDK	<a href="http://www.infineon.com/matrikonopc">www.infineon.com/matrikonopc</a>
Embedded coder library for Matlab	<a href="http://www.infineon.com/matlab">www.infineon.com/matlab</a>
XMC 底层驱动库	<a href="http://www.infineon.com/xmclib">www.infineon.com/xmclib</a>
XMC 方案和技术介绍视频	<a href="http://www.infineon.com/xmc-mediacyenter">www.infineon.com/xmc-mediacyenter</a>
XMC4800 产品	<a href="http://www.infineon.com/xmc4800">www.infineon.com/xmc4800</a>
XMC EtherCAT 产品和技术介绍	<a href="http://www.infineon.com/ethercat">www.infineon.com/ethercat</a>
XMC Arduino 产品介绍	<a href="http://www.infineon.com/arduino">www.infineon.com/arduino</a>
XMC Link	<a href="http://www.infineon.com/xmclink">www.infineon.com/xmclink</a>
XMC Flasher	<a href="http://www.infineon.com/xmcflasher">www.infineon.com/xmcflasher</a>
DAVE 主页	<a href="http://www.infineon.com/xmcflasher">www.infineon.com/xmcflasher</a>