



# 基于 MDK 开发的 TencentOS-Tiny Software Pack

## TencentOS-Tiny software package based on MDK development

2021/9/3

东南大学

导师：汪礼超

学员：崔林威

Email: 1797878653@qq.com

<https://github.com/OpenAtomFoundation/TencentOS-tiny>

# 目录

1、ARM 软件包介绍 .....	3
1.1 软件包简介.....	3
1.2 软件包开发.....	3
1.2.1 软件包开发过程.....	3
1.2.2 PDSC 文件的编写.....	4
1.2.3 生成软件包.....	7
2、TencentOS-tiny 软件包 .....	10
2.1 软件包内容.....	10
2.2 软件包安装.....	10
3、软件包测试 .....	13
3.1 ARM 内核移植 TencentOS tiny 软件包.....	13
3.2 STM32 不依赖裸机工程移植 .....	15
3.3 单片机裸机工程移植.....	17
4、总结 .....	22
5、开发参考 .....	23
6、附录-移植配置参考 .....	24
6.1 MDK5.14 版本移植到 ARM 内核 .....	24
6.1.1 Cortex-M0 内核移植.....	24
6.1.2 Cortex-M0+内核移植.....	24
6.1.3 Cortex-M3 内核移植.....	25
6.1.4 Cortex-M4 内核移植.....	25
6.1.5 Cortex-M7 内核移植.....	26
6.2 MDK5.14 版本移植到基于 ARM 内核的芯片 .....	26
6.2.1 移植到 stm32f103c8 芯片.....	26
6.2.2 移植到 stm32f767igt 芯片 .....	27
6.3 MDK5.30 和 MDK5.35 版本移植（Cortex-M0+、0、3、4、7 内核和芯片） .....	27
6.4 MDK5.30 和 MDK5.35 版本移植（Cortex-M23、33） .....	28
6.4.1 Cortex-M23 内核移植.....	28
6.4.2 Cortex-M33 内核移植.....	29

# 1、ARM 软件包介绍

## 1.1 软件包简介

在进行嵌入式软件开发时，ARM 为我们提供了软件包功能，能够将软件算法等模块进行集成封装，从而方便第三方用户使用。ARM 软件包能够为微控制器设备和开发板提供支持，包含软件组件（Software Component）如驱动程序和中间件，还可以包含示例项目和代码模板等，主要有以下类型的软件包：

(1) 器件系列包（Device Family Pack）：由硅供应商或工具供应商生成，为特定的目标微控制器创建软件应用提供支持；

(2) 板级支持包（Board Support Pack）：由电路板供应商发布，为安装在电路板上的外围硬件提供软件支持。

(3) CMSIS 软件包：由 ARM 提供，包括对 CMSIS 核心、DSP 和 RTOS 的支持；

(4) 中间件包（Middleware Pack）：由芯片供应商、工具供应商或第三方创建；通过提供对常用软件组件（如软件堆栈、特殊硬件库等）的软件集成，从而减少开发时间；

(5) 内部组件（In-house components）：由工具用户开发，用于内部或外部分发。

软件组件包括以下几部分：

(1) 源代码、库、头文件/配置文件和文档；

(2) 完整的示例项目，展示了软件组件的使用，可以下载并在评估硬件上执行；

(3) 代码模板，方便使用软件组件。

一个完整的软件包是一个 ZIP 文件，包含所有需要的软件库和文件，以及一个包含软件包所有信息的包描述文件（PDSC 文件），软件包的结构是在 CMSIS 中定义的 (<http://www.keil.com/CMSIS/Pack>)。

## 1.2 软件包开发

### 1.2.1 软件包开发过程

软件包的开发过程相当于完成了一项产品的制作，因此引入产品生命周期管理（PLM）的概念，PLM 包括以下四个阶段：（1）概念的产生，基于软件包需求进行产品定义，并创建第一个功能原型；（2）设计，根据技术特征和要求，进行原型测试和产品的实施，通过广泛的测试验证产品的功能与规格；（3）发布，产品被制造出来并推向市场；（4）服务，对产品的维护，包括对客户的支持，最后不断优化，结束产品的周期。

在制作软件包时，主要面临以下几个过程：

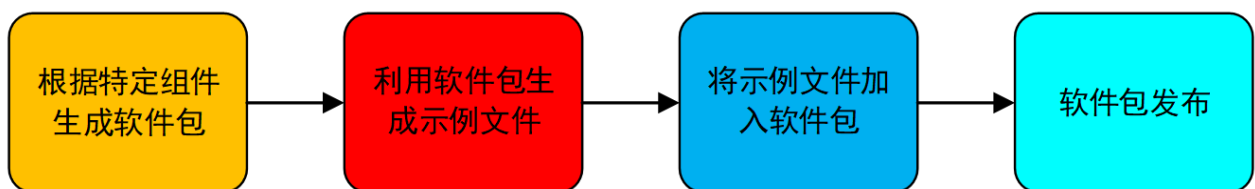


图 1.1 软件包开发流程

首先，根据特定组件生成软件包即根据需求将相应的头文件、库文件等软件组件利用 PDSC 文件进行组织，在组织完成后即可利用软件包生成工具生成对应版本的软件包，然后对新生成的软件包进行测试，给出示例测试程序，再将其包含如 PDSC 文件中，最后经测试完成后生成最终的软件包。

### 1.2.2 PDSC 文件的编写

PDSC 文件是基于可扩展标记语言（XML）进行编写的，能够将软件包包含的各个模块按照特定的格式组织起来，接下来按照 PDSC 文件的结构对文件的编写进行详细介绍：

首先是 PDSC 文件的开头，前两句是声明为 XML 格式，它是在 MDK 中的 PACK.xsd 文件定义的，所以不用修改；<name>和<vendor>标签定义了软件包的基本内容，也用于 PACK 文件的文件名，故该 PDSC 文件应命名为 Tencent.TencentOS-tiny.pdsc；<description>标签描述了软件包的信息，它将显示在包安装程序中；<url>标签可以包含一个带有软件包下载链接的网址，方便用户下载；<license>标签包含了用户使用该软件包时需要遵守的协议，<supportContact>标签表示软件包的支持人员联系方式，可以提供一个电子邮件地址或网页 URL。如图 1.2 为下列代码对应的软件包界面。

```
<?xml version="1.0" encoding="utf-8"?>
<package schemaVersion="1.0" xmlns:xs=http://www.w3.org/2001/XMLSchema-instance
xs:noNamespaceSchemaLocation="PACK.xsd">
<name>Tencent</name>
<description>Description of your pack</description>
<vendor>TencentOS-tiny</vendor>
<url>https://github.com/OpenAtomFoundation/TencentOS-tiny</url>
<license>LICENSE.txt</license>
<supportContact>...</supportContact>
```

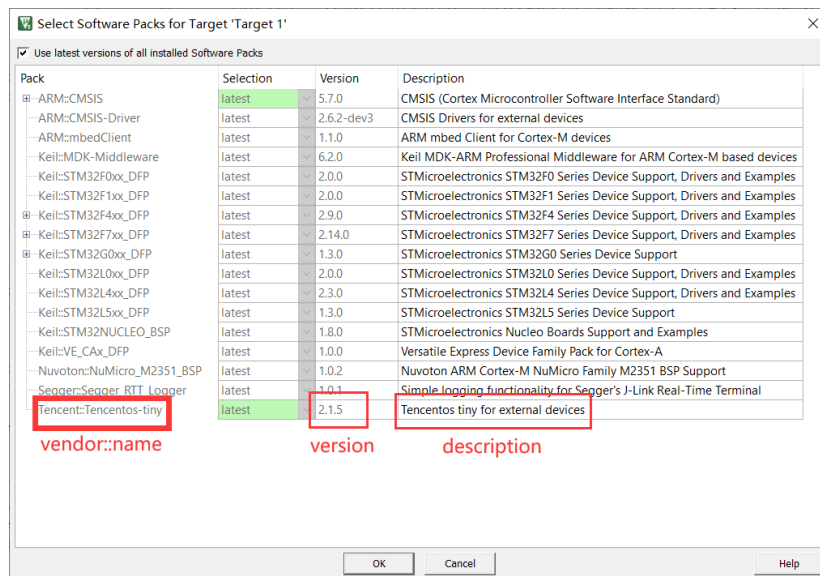


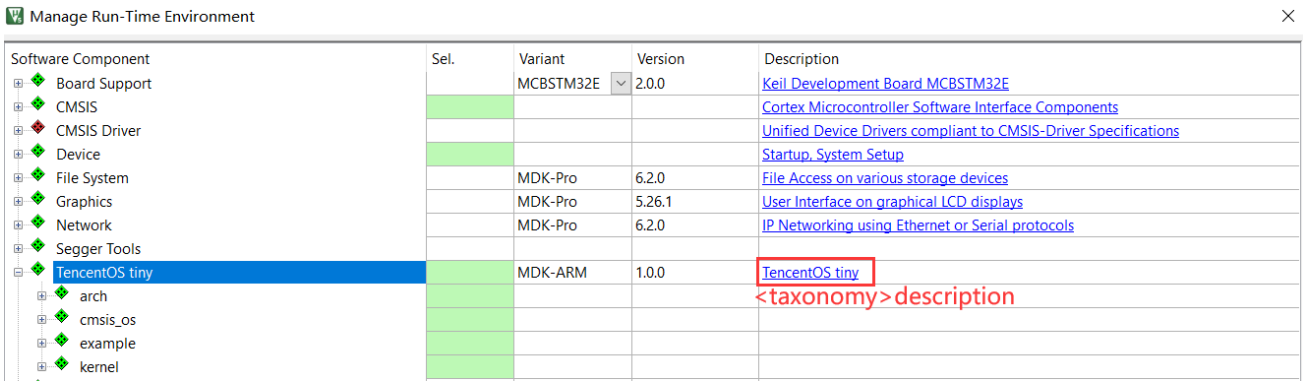
图 1.2 程序对应的软件包

接下来是 PDSC 文件的各个模块，<releases>标签定义了软件包的版本，开发者可以在版本更新时在此进行标注，从而在生成软件包时，系统会自动生成最新版本的软件包；

```
<releases>
<release version="1.0.1">
    Sep/3/2021, version name
</release>
<release version="1.0.0">
    Sep/1/2021, version name
</release>
</releases>
```

<taxonomy>标签用于定义每个组件的 description,如图 1.3 所示,通过下列代码中的 Cclass、Cgroup 和 Csub 来确定 description 所在的位置, doc 用于指定 description 文件 (也可以不加),然后添加 description 的名字。

```
<taxonomy>
  <description Cclass="TencentOS tiny" Cgroup="xx" Csub="xx" doc="examples/index.html">TencentOS
  tiny</description>  <!--添加网页功能 -->
</taxonomy>
```



Software Component	Sel.	Variant	Version	Description
Board Support		MCBSTM32E	2.0.0	Keil Development Board MCBSTM32E
CMSIS				Cortex Microcontroller Software Interface Components
CMSIS Driver				Unified Device Drivers compliant to CMSIS-Driver Specifications
Device				Startup, System Setup
File System		MDK-Pro	6.2.0	File Access on various storage devices
Graphics		MDK-Pro	5.26.1	User Interface on graphical LCD displays
Network		MDK-Pro	6.2.0	IP Networking using Ethernet or Serial protocols
Segger Tools				
TencentOS tiny		MDK-ARM	1.0.0	TencentOS tiny <taxonomy>description
arch				
cmsis_os				
example				
kernel				

图 1.3 <taxonomy>标签

< keywords >标签定义了软件包的关键词,在 ARM 官网下载软件包时可利用关键词搜索到需要的软件包。

```
<keywords >
  <keyword>Tencent</keyword>
</keywords >
```

< requirements >标签定义了软件包的关联安装需求,即在安装本软件包时,还需要在网上安装其他包 (网址: [MDK5 Software Packs \(keil.com\)](http://MDK5 Software Packs (keil.com))),例如下面的定义则需要我们安装 ARM 的 CMSIS5.7.0 软件包。

```
<requirements>
  <packages>
    <package vendor="ARM" name="CMSIS" version="5.7.0"/>
  </packages>
</requirements>
```

接下来是<conditions>标签,该标签在设计<components>时使用,用以说明软件包中各个组件的依赖关系,即使用本组件还需要选择其他组件。在该标签下可以定义多个 condition,每个 condition 可以定义多个条件。其中<conditions id>表示条件名,<description>为条件信息,然后便是定义的条件,其中<accept>表示该条件时可选的,当同时存在多个<accept>时,用户需要至少满足其中一个条件才可以使用;<require>表示该条件是必选的,否则便无法使用相应的组件。在条件内部,包含一些特定的指示性语法,如果在设计<component>的时候开发者选择了名为 Cortex\_M0 的条件,那么用户在使用该<component>时,则需要遵守条件要求:其中<accept Dvendor="ARM:82" Dname="ARMCM0"/>表示用户选择 ARM-Cortex M0 内核时才会触发该条件,<require condition="condition id"/>为条件嵌套,表示用户还需要满足嵌套条件对应的要求,<require Cclass="TencentOS tiny" Cgroup="kernel" Csub="core"/>表示用户还需要选择 core 组件。

```
<conditions>
  <condition id="Cortex_M0">
    <description> Cortex-M0</description>
```

```
<accept Dvendor="ARM:82" Dname="ARMCM0"/>
<require condition="condition id "/>
<require Tcompiler="ARMCC"/>
<require Cclass="TencentOS tiny" Cgroup="kernel" Csub="core"/>
</condition>
<condition id=" condition_2 ">
  <description></description>
  <!-- 第二个 condition 的内容 -->
</condition>
</conditions>
```

然后是<components>标签，该标签描述了软件包包含的所有文件，在编写该标签下的程序时，需要按照文件类别将文件进行划分，在下列代码中，定义了一个 Keil:: TencentOS tiny:: arch::arch 的<component>，<description>为该组件的信息，具体如图 1.4 所示。

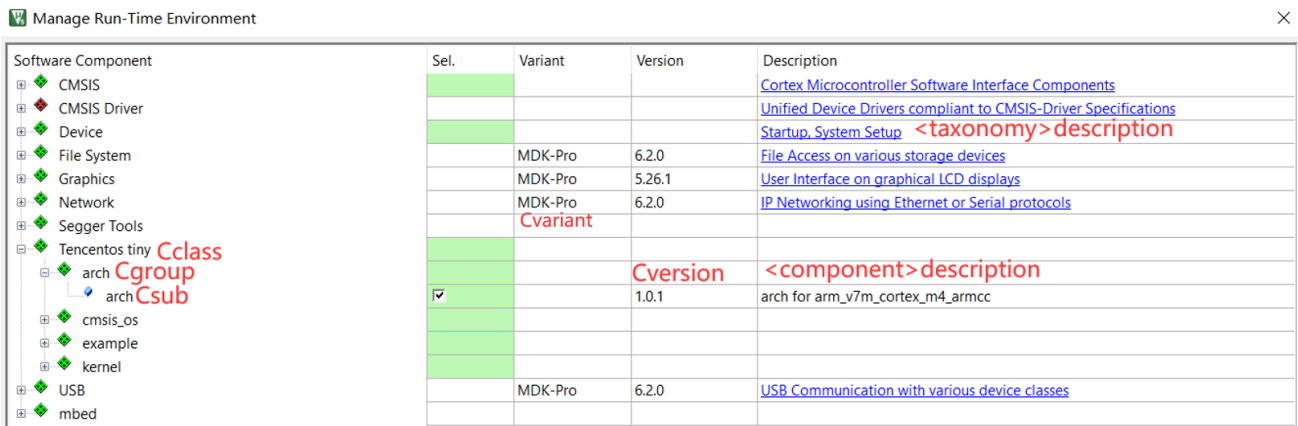


图 1.4 <component>定义界面

```
<components>
  <component Cvendor="Keil" Cclass="TencentOS tiny" Cgroup="arch" Csub="arch" Cversion="1.0.1"
    condition=" condition id">
    <description> description </description>
    <files>
      <file category="doc" name="Documentation/General/html/driver_I2C.html"/>-->
      <file category="include" name="arch/arm/arm-v7m/common/include/" />
      <file category="header" name="arch/arm/arm-v7m/cortex-m0+/armcc/port.h"/>
      <file category="header" name="arch/arm/arm-v7m/cortex-m0+/armcc/port_config.h" attr="config"
        version="1.1.0"/>
      <file category="source" name="arch/arm/arm-v7m/common/tos_cpu.c"/>
    </files>
  </component>
</components>
```

condition=" condition id"即为上述介绍的<condition>标签，从而用户在使用该组件时，还需要满足 condition 所要求的依赖组件。另外，在定义某一个<component>时，需要按照上面程序的<files>...</files>语法进行文件添加，其中 file category 的定义如表 1-1 所示，在 name 中可以添加文件路径和具体的某个文件，在软件包中，我们添加的文件默认是不可编辑的，为了方便用户对文件进行配置，我们需要添加 attr="config"属性，并可通过 version 更新不同版本的文件。

表 1-1 file category 定义

category	含义
doc	文件，可以是网页或者其他链接
include	包含某一个路径下的所有头文件
header	包含某路径下的具体头文件
source	.c 源文件

为了使我们设计的软件包能够适配不同的内核，即在使用软件包时与用户 ARM 核不一致的文件都不出现，可以按如下步骤进行<files>的添加：（1）在 condition 条件中，加入<require Dvendor="ARM:82" Dname="ARMCM0"/>这样的程序，该程序表示需要用户选择 ARM Cortex-M0 内核才会触发该条件；（2）在添加<files>时，我们可以将针对不同内核，同一类型文件的 Cgroup 和 Csub 保持同样的名字，并添加上（1）中定义的 condition，这样用户选择不同内核时，将只会出现与用户内核一致的文件。

同时，如果需要在 PDSC 文件中定义多个软件包，可以采用下列代码结构，其中每一个<bundle>标签定义了一个软件包。

```
<components>
  <bundle Cbundle="MDK-ARM" Cclass="TencentOS tiny" Cversion="1.0.0">
    <description>TencentOS tiny</description>
    <doc>examples/index.html</doc>    <!--添加网页功能 -->
    <component
      <!-- 组件内容 -->
    </component>
  </bundle>
  <bundle Cbundle="MDK-ARM" Cclass="TencentOS tiny" Cversion="1.0.0">
    <description>TencentOS tiny</description>
    <doc>examples/index.html</doc>
    <component
      <!-- 组件内容 -->
    </component>
  </bundle>
</components>
```

另外，PDSC 文件还可以包含<devices>、<apis>、<boards>和<examples>，这些为 ARM 公司或者其他器件、开发板厂商提供，为针对器件、api 库文件、板级和相应的示例文件，具体可以参阅 ARM CMSIS 的软件包。

最后，PDSC 文件后还需要在最后加上</package>，表示该文件的结束，从而完成 PDSC 文件的编写。

### 1.2.3 生成软件包

在完成 PDSC 文件的编写后，为了生成最终的软件包，还需要准备如图 1.5 所示的 3 个文件，其中 PackChk.exe 用于验证软件包包含的文件是否都存在，即是否完整；gen\_pack.bat 为 Windows 批处理文件，需要我们对文件中的路径进行修改，并用于生成软件包；PACK.xsd 是 schema，主要用来制定 XML 规范，用以验证我们编写的 PDSC 文件。另外，还需要准备 7-Zip File Manager 软件，用于对文件进行压缩，制作集成的软件包。



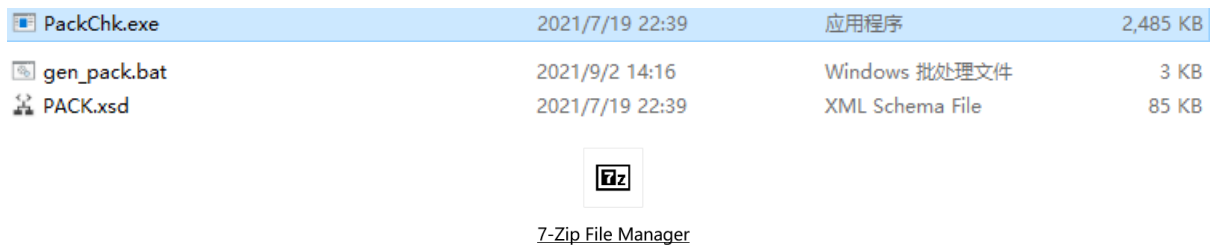


图 1.5 生成软件包所需的软件配置

首先利用记事本或 Notepad++ 打开 gen\_pack.bat, 对以下几个地方需要修改, 如表 1-2 所示:

```
SET ZIPPATH=C:\Program Files\7-Zip
SET RELEASE_PATH=..\Local_Release
SET PACK_VENDOR=Tencent
SET PACK_NAME=TencentOS-tiny
SET PACK_FOLDER_LIST=arch osal kernel examples
SET PACK_FILE_LIST=%PACK_VENDOR%.%PACK_NAME%.pdsc README.md LICENSE.txt
```

表 1-2 gen\_pack.bat 修改内容

代码	含义
SET ZIPPATH	7-Zip File Manager 软件的安装路径
SET RELEASE_PATH	生成的软件包路径, 为相对路径
SET PACK_VENDOR	PDSC 文件中的<vendor>标签
SET PACK_NAME	PDSC 文件中的<name>标签
SET PACK_FOLDER_LIST	软件包包含文件所在路径
SET PACK_FILE_LIST	README.md LICENSE.txt 所在路径

修改完毕 gen\_pack.bat 后, 便可以制作软件包了, 首先利用 cmd 打开电脑的命令行界面, 执行 cd 命令转到 gen\_pack.bat 所在的路径, 然后输入 gen\_pack.bat 点击 enter, 如图 1.6 所示, gen\_pack.bat 会按照顺序压缩文件, 然后读取 PDSC 文件, 检查数据完整性和文件依赖是否完整, 然后生成软件包, 当提示 gen\_pack.bat completed sucessfully 后就完成了软件包的创建。

```
命令提示符
Microsoft Windows [版本 10.0.19042.1165]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\lw_cui>cd C:\Users\lw_cui\Desktop\Tencentos-Tiny\Utilities

C:\Users\lw_cui\Desktop\Tencentos-Tiny\Utilities>gen_pack.bat
removing ..\Local_Release

Delete previous generated HTML files

Generate New HTML Files
'doxygen' 不是内部或外部命令, 也不是可运行的程序
或批处理文件。

已复制 1 个文件。
已复制 1 个文件。
已复制 1 个文件。
复制了 197 个文件
复制了 35 个文件
复制了 90 个文件
复制了 57 个文件
PackChk.exe 1.3.53 (19.07.2021 22:39)
Copyright (C) 2012-2016 ARM Ltd and ARM Germany GmbH. All rights reserved.

M362: Also suppressing Messages M502 and M504

Phase1: Read PDSC files

Phase2: Static Data & Dependencies check
```



```
Phase3: RTE Model based Data & Dependencies check
Found 0 Error(s) and 6 Warning(s).
Creating pack file ...
Packaging complete
Removing temporary files and folders
gen_pack.bat completed successfully
```

图 1.6 软件包生成界面

此时在 Local\_Release 路径下，可以看到生成的软件包。


2、软件包源文件 > Local_Release		搜索"Local_Release"	
名称	修改日期	类型	大小
 Tencent.TencentOS-tiny.1.0.0.pack	2021/9/19 17:42	uVision Software Pack	683 KB

图 1.7 软件包

## 2、TencentOS-tiny 软件包

腾讯物联网操作系统（TencentOS tiny）是腾讯面向物联网领域开发的实时操作系统，具有低功耗，低资源占用，模块化，可裁剪等特性。TencentOS tiny 提供了最精简的 RTOS 内核，内核组件可裁剪可配置，可灵活移植到多种终端 MCU 上。而且，基于 RTOS 内核，提供了 COAP/MQTT/TLS/DTLS 等常用物联网协议栈及组件，方便用户快速接入腾讯云物联网通信 IoT Hub。同时，TencentOS tiny 为物联网终端厂家提供一站式软件解决方案，方便各种物联网设备快速接入腾讯云，可支撑智慧城市、智能水表、智能家居、智能穿戴、车联网等多种行业应用。

因此，为了有效减少开发人员在移植 TencentOS tiny 到 ARM 内核单片机上的开发时间，本文基于 MDK 完成了第三方 TencentOS Tiny pack 和软件包的封装，能够使用 MDK pack 直接生成适合不同 MCU 的 TencentOS Tiny 工程。

### 2.1 软件包内容

结合 TencentOS tiny 的算法架构，本文设计的软件包包括如表 2-1 所示的内容：

表 2-1 软件包内容

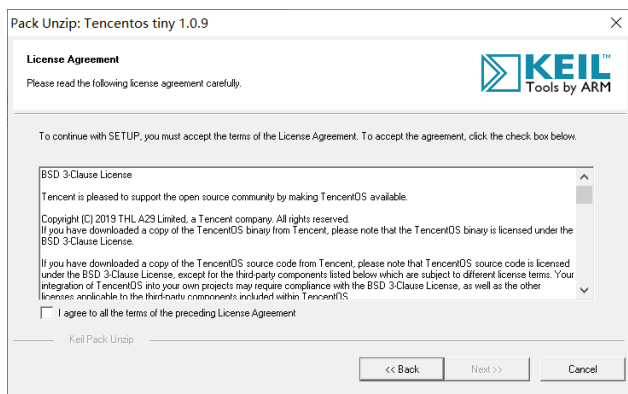
内容		功能
arch		包括 TencentOS-tiny\arch\arm 下内核为 Cortex-M0+、Cortex-M0、Cortex-M3、Cortex-M4、Cortex-M7、Cortex-M23、Cortex-M33 的 arch 文件
kernel		包括 TencentOS-tiny\kernel 下的 core、hal 路径中的文件、tos_config 文件
cmsis_os		对应 TencentOS-tiny\osal\cmsis_os 的文件
example	helloworld_main	用于测试软件包的 main 文件
	mcu_it.c	移植软件包时需要按照该文件对中断函数进行修改
	mcu_platform.h	用户可在此文件在添加对应单片机的头文件

软件包具有以下功能：

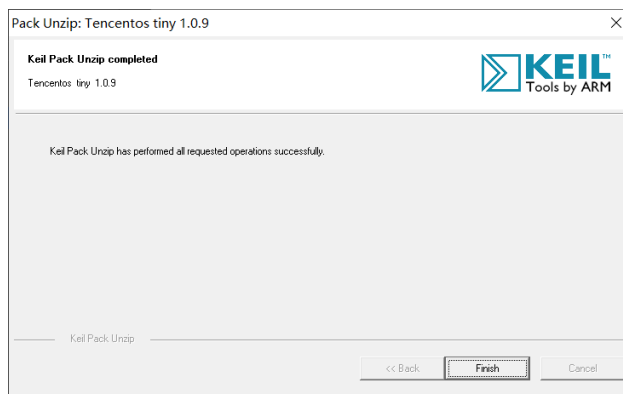
- （1）软件包针对 ARMCortex-M0+、Cortex-M0、Cortex-M3、Cortex-M4、Cortex-M7、Cortex-M23 和 Cortex-M33 内核进行了 TencentOS tiny 软件的封装，用户在安装软件包后能够快速将 TencentOS tiny 相应内核的 Keil 工程中；
- （2）软件包能够自动适应用户所选的内核，arch 文件能够根据内核自动显示，从而方便用户使用；
- （3）用户在勾选一个组件时，软件包会自动提示还需要勾选其他模块，并可利用界面中的 Resolve 一键勾选，防止遗漏；
- （4）用户可自主修改对应内核的 tos\_config 文件，对 TencentOS tiny 的功能进行裁剪。

### 2.2 软件包安装

接下来介绍 Tencent.TencentOS-tiny 软件包的安装，首先双击图 1.5 中的软件包，然后进入安装界面，如图 2.1(a)，点击 I agree to all the terms of the preceding License Agreement，再点击 next 进行安装，安装完成界面如图 2.1(b)所示；



(a)



(b)

图 2.1 安装界面

此时软件包已经安装到 Keil 5 之中，打开 Keil 5 软件，并点击 Pack Installer 图标，可以进行不同软件包版本的安装与移除：

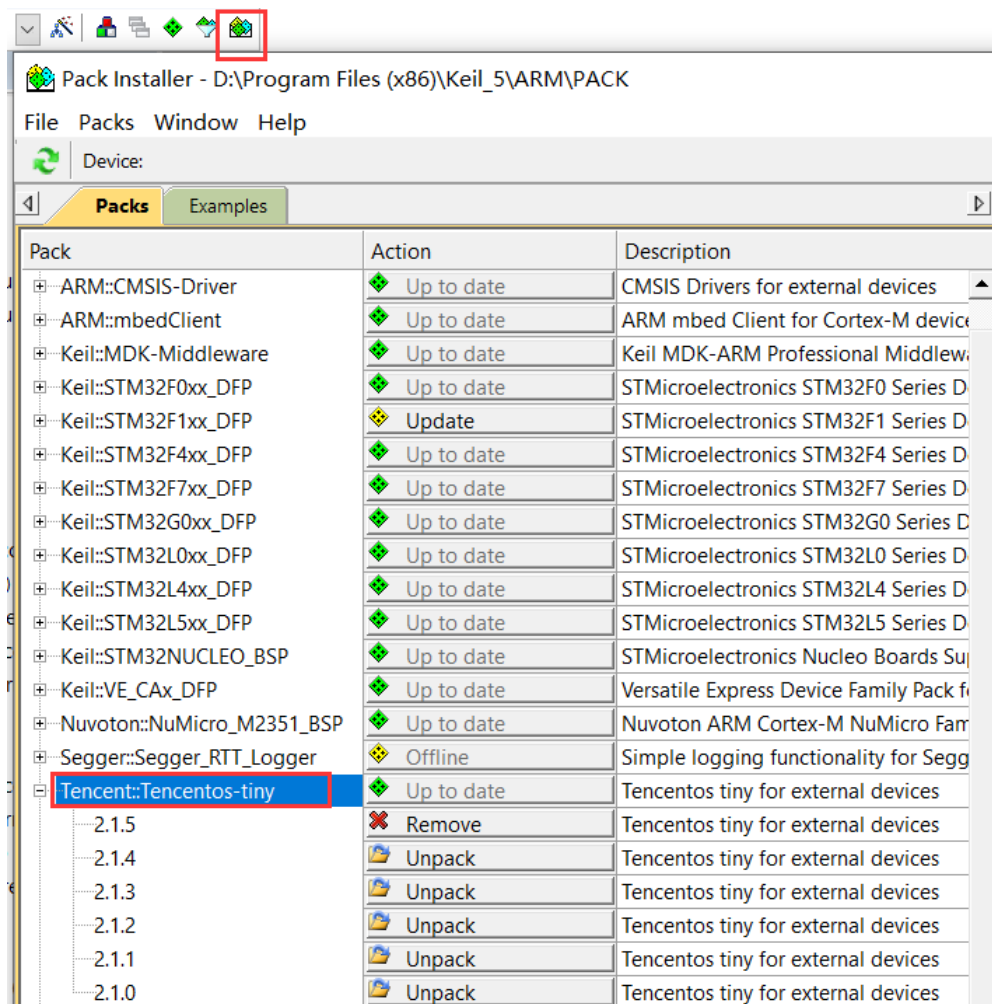


图 2.2 Pack Installer 界面

接下来就可以安装 Tencent.TencentOS-tiny 软件包中的组件，点击 Manage Run-Time Environment 图标，对需要从软件包中移植的文件进行勾选，如图 2.3 所示，如果有依赖可以点击 Resolve 进行一键安装。

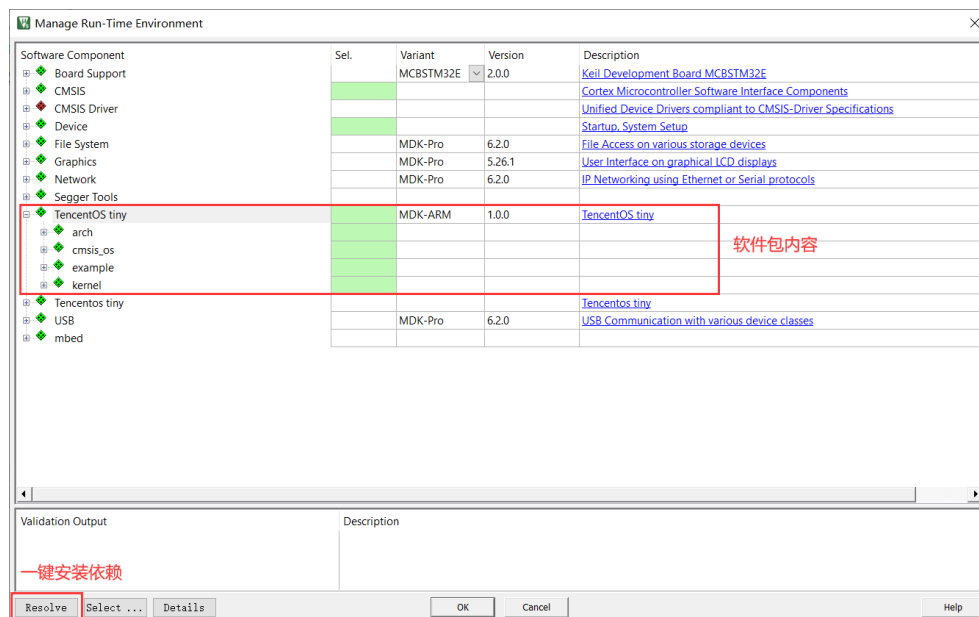


图 2.3 Manage Run-Time Environment 界面

### 3、软件包测试

#### 3.1 ARM 内核移植 TencentOS tiny 软件包

首先在 [MDK5 Software Packs \(keil.com\)](#) 下载安装 ARM CMSIS-5.7.0 软件包，以便在不同内核下测试本软件包。

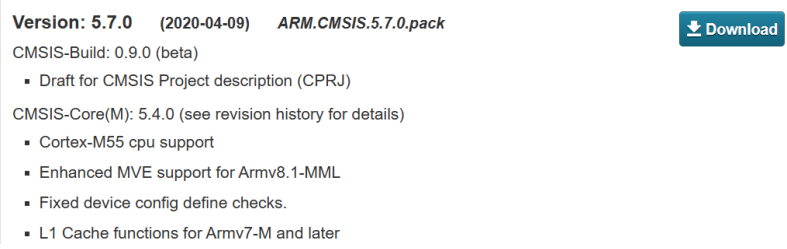


图 3.1 ARM CMSIS-5.7.0 软件包

在安装完软件包后，以 ARM Cortex-M3 内核为例对软件包进行移植，并进行编译，首先利用 Keil5-5.30 版本软件新建工程，并选择 ARMCM3，如图 3.2 所示，然后按照图 3.3 勾选相应的 TencentOS-tiny 组件和 Cortex-M3 内核文件，可以看到 arch 和 tos\_config 都已经根据内核进行了自动适配。

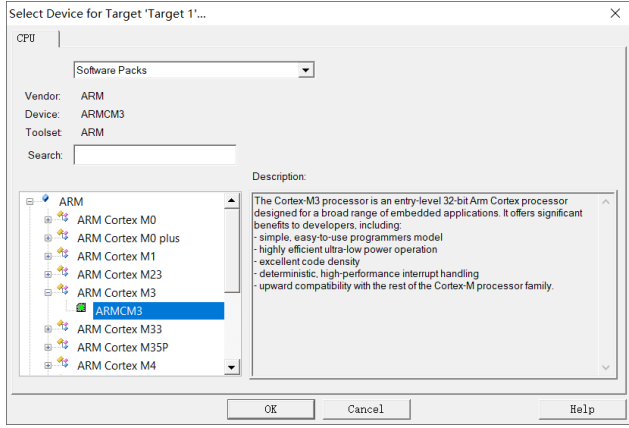


图 3.2 勾选内核

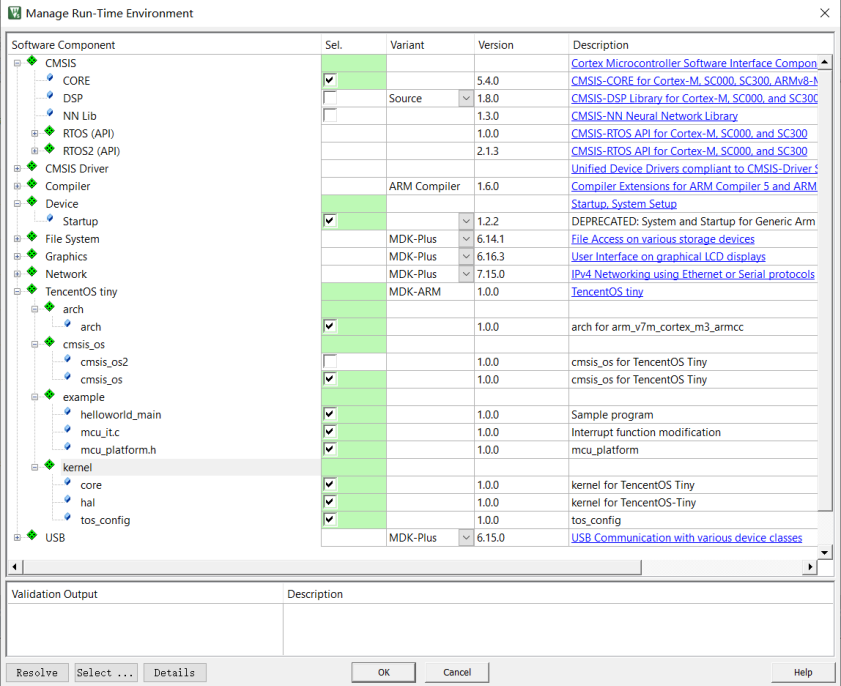


图 3.3 勾选组件

接下来点击 Options for target, 选中默认编译版本 5, 然后选择 C99 mode。

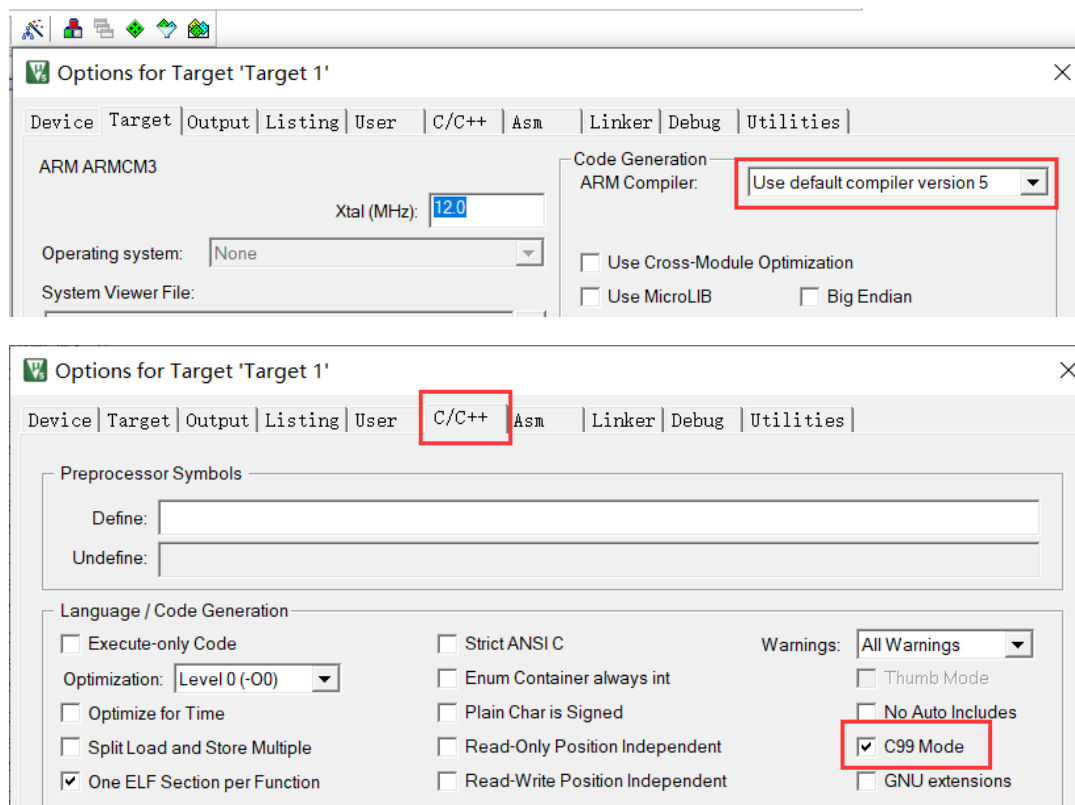


图 3.4 Options for target

然后在 mcu\_platform.h 中添加#include "ARMCM3.h"和#include "core\_cm3.h"头文件。

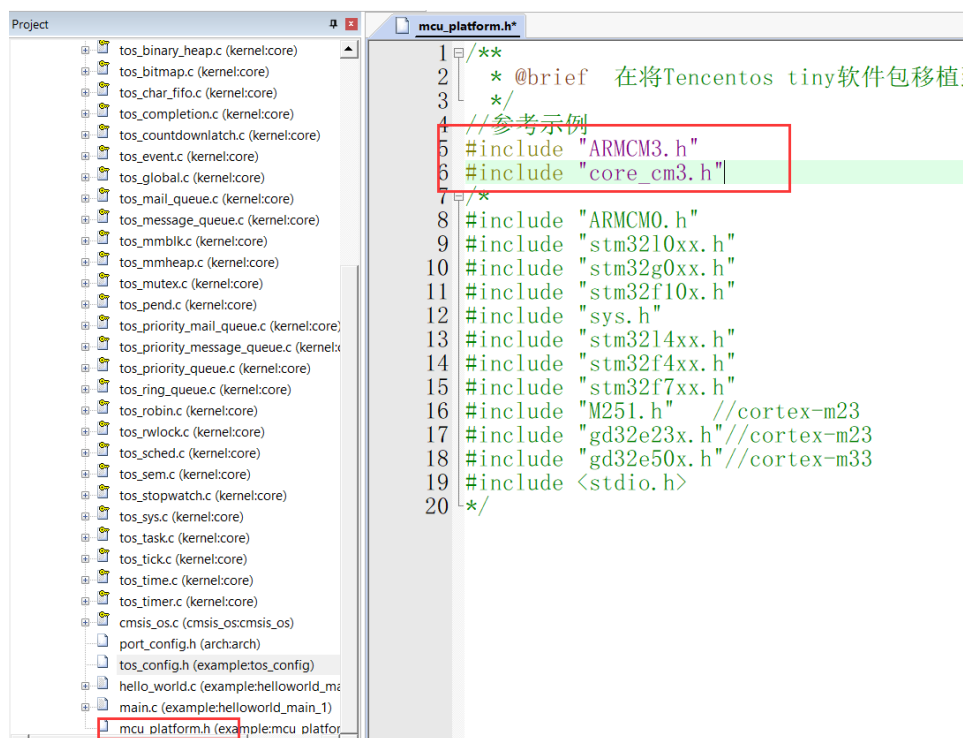


图 3.5 添加对应内核的头文件

最后点击 Build 图标进行测试, 如图 3.6 所示:

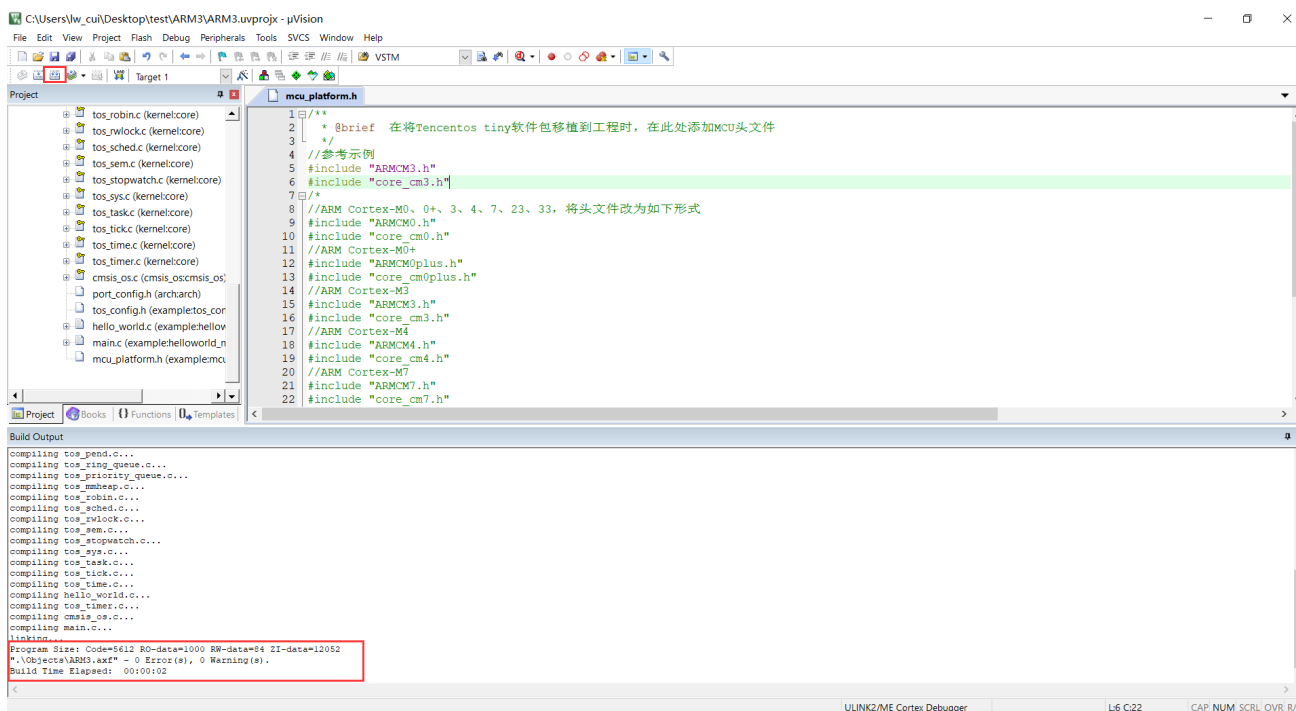


图 3.6 编译测试

与之类似，如果要在 ARM Cortex-M4 内核下对本软件包进行测试，只需要在上述步骤修改 mcu\_platform.h 中的头文件为 #include "ARMCM4.h" 和 #include "core\_cm4.h" 即可，如果是其他内核则对应修改头文件。

### 3.2 STM32 不依赖裸机工程移植

接下来选取具体单片机芯片，进行软件包的测试，按照以下步骤：

在网站 [MDK5 Software Packs \(keil.com\)](http://www.keil.com/MDK5_Software_Packs) 上下载 STM32F1 的软件支持包，如图 3.7 所示，并进行安装。

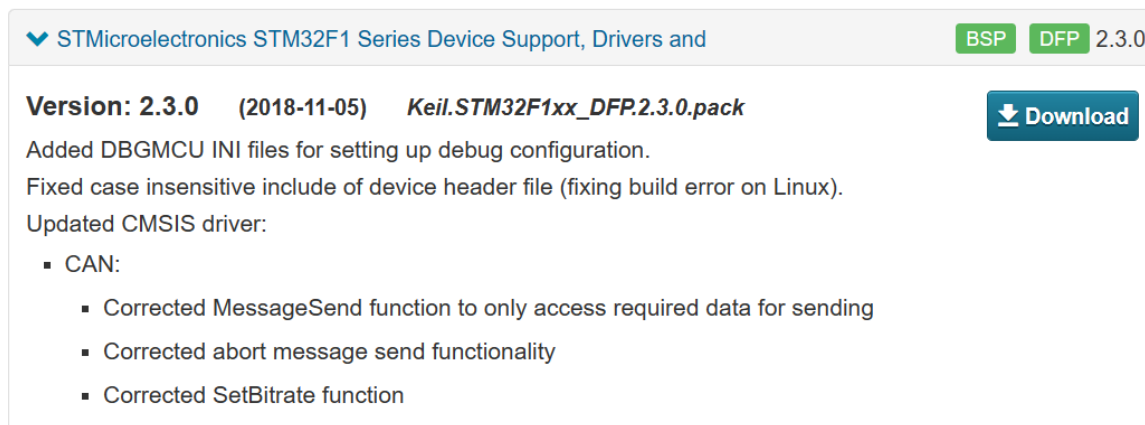


图 3.7 STM32 软件支持包

新建工程，选择芯片为 STM32F103C8，如图 3.8 所示，然后点击 ok，按照图 3.9 所示，选择 TencentOS-tiny 软件包的组件和 STM32 的启动文件。



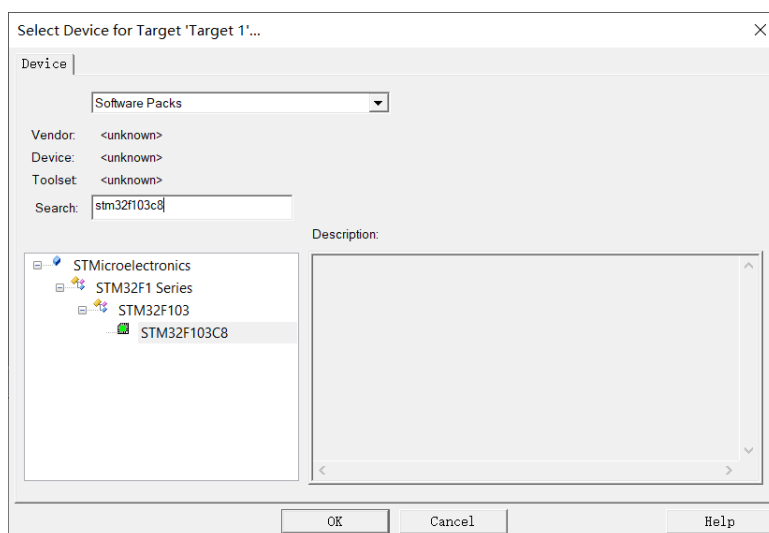


图 3.8 选择 STM32F103C8 芯片

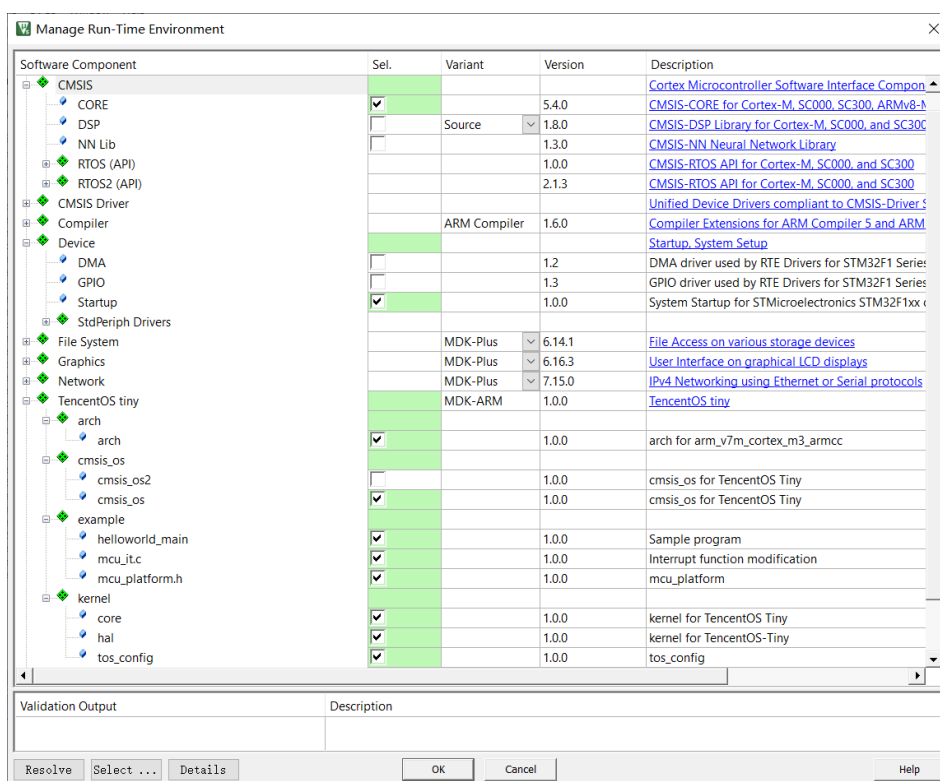
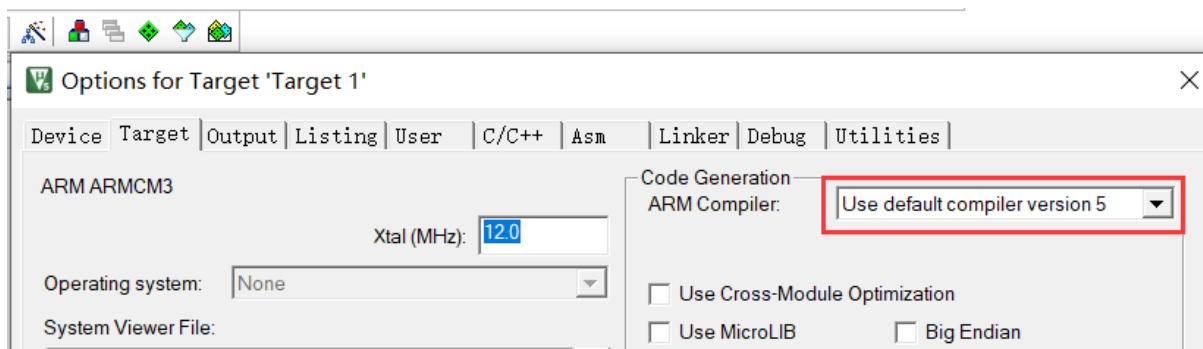
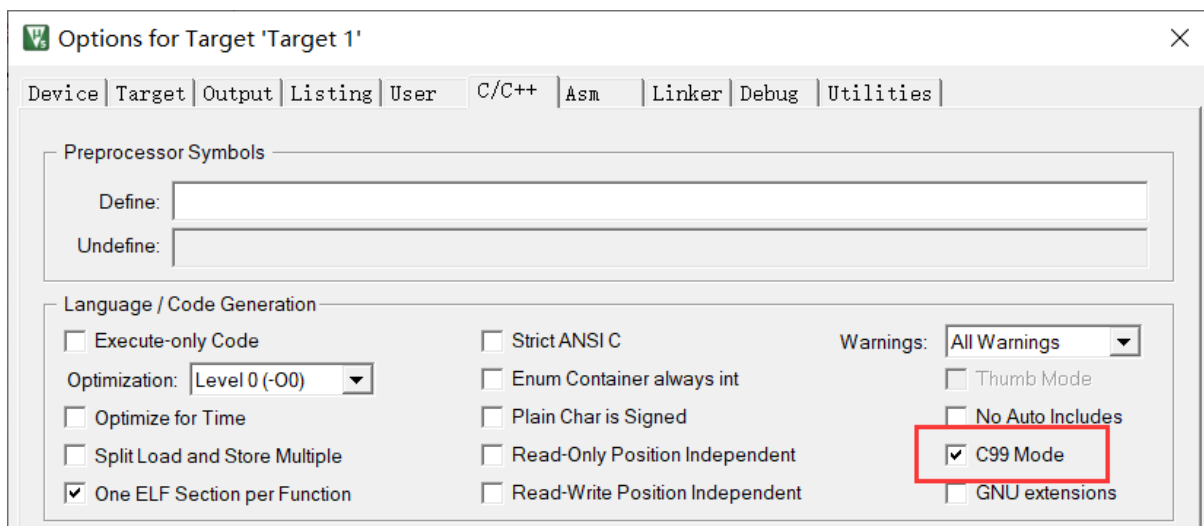


图 3.9 选择组件

然后按照图 3.10(a)所示勾选编译版本 5，并选择 C99mode。



(a)



(b)

图 3.10 软件设置

然后按照图 3.11，在 mcu\_platform.h 中添加以下头文件：

```
#include "stm32f10x.h"
#include "core_cm3.h"
#include "system_stm32f10x.h"
```

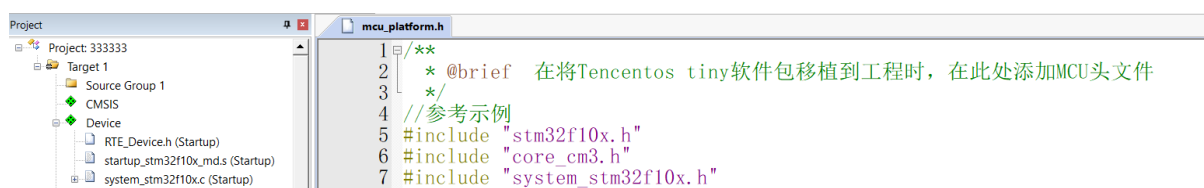


图 3.11 修改 mcu\_platform.h

最后点击 Build 编译，没有报错则移植成功。

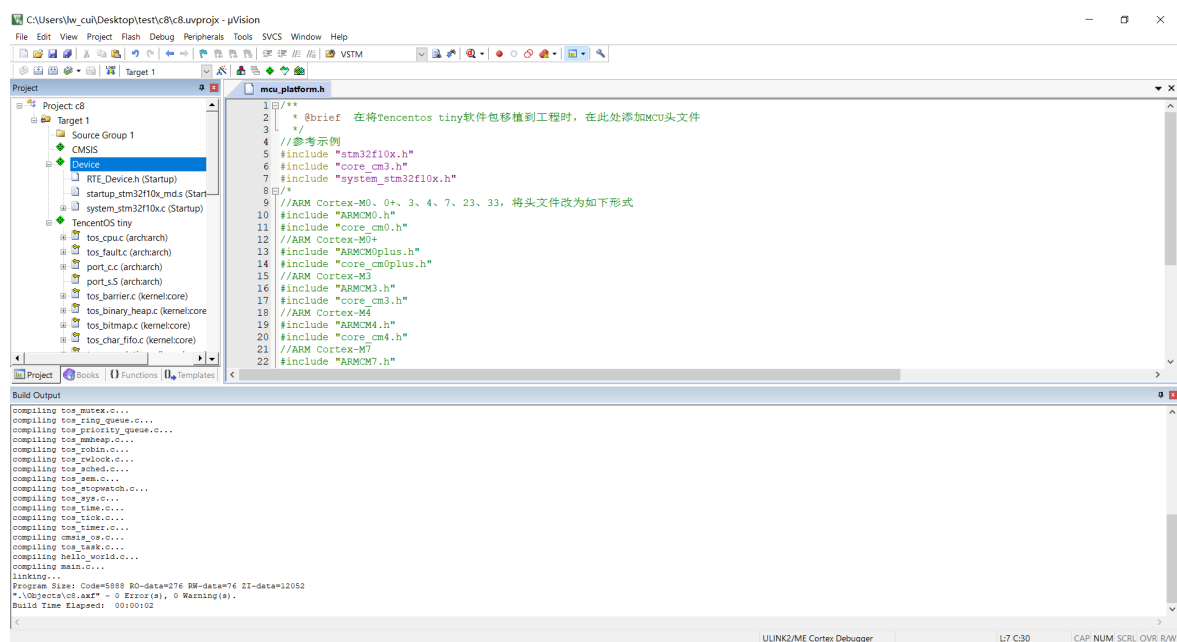


图 3.12 编译界面

### 3.3 单片机裸机工程移植

最后对单片机开发板进行测试，以正点原子探索者 STM32F407ZGT6 为例介绍

TencentOS-tiny 软件包的移植。

(1) 如下图为软件包勾选的内容：

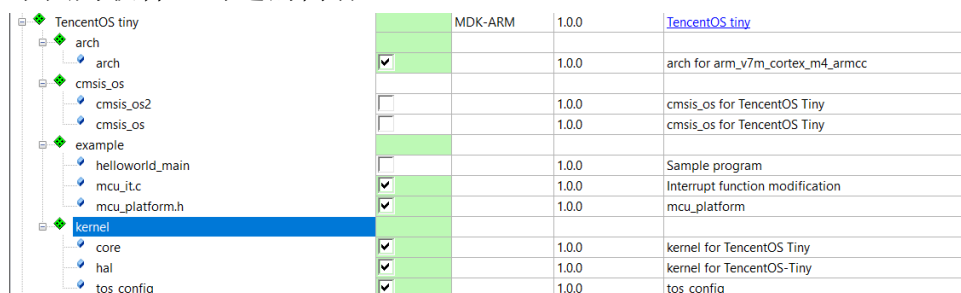


图 3.13 软件包组件勾选

在正点原子探索者 STM32F407ZGT6 裸机工程模板中移植软件包后的界面如图 3.14 所示：

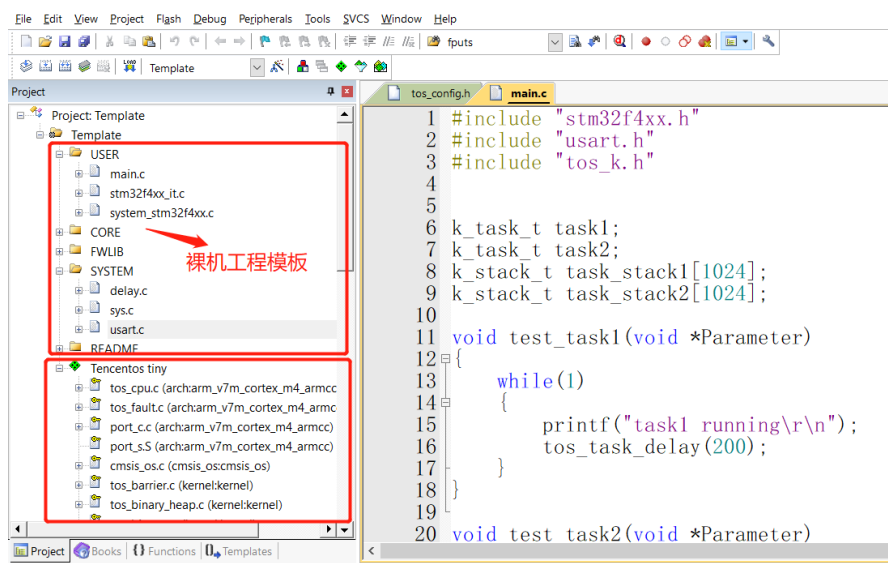


图 3.14 移植界面

(2) 然后按照 mcu\_it.c 对 stm32f4xx\_it.c 中的 PendSV\_Handler()函数和 SysTick\_Handler()函数进行修改，如下图所示，注释 stm32f4xx\_it.c 中的 PendSV\_Handler()函数，并修改 SysTick\_Handler()函数。

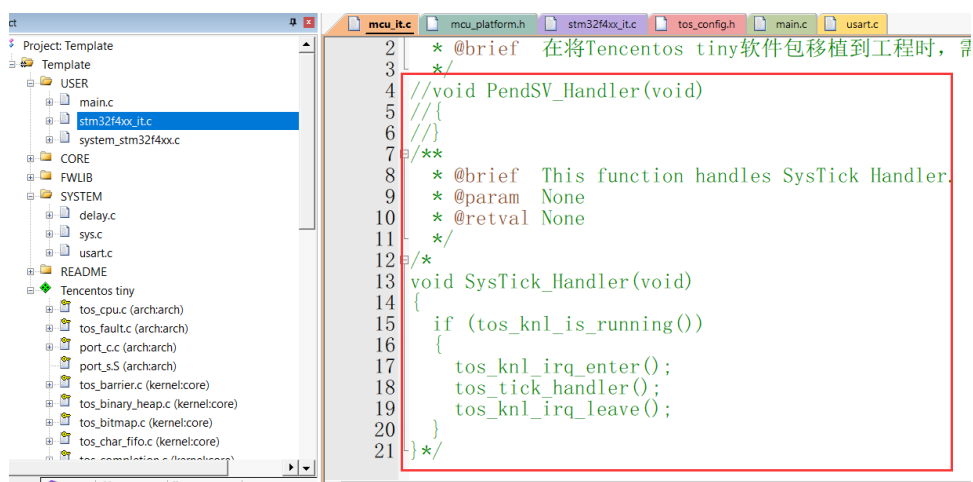


图 3.15 函数修改

(3) 对 mcu\_platform.h 进行修改，添加#include "stm32f4xx.h"

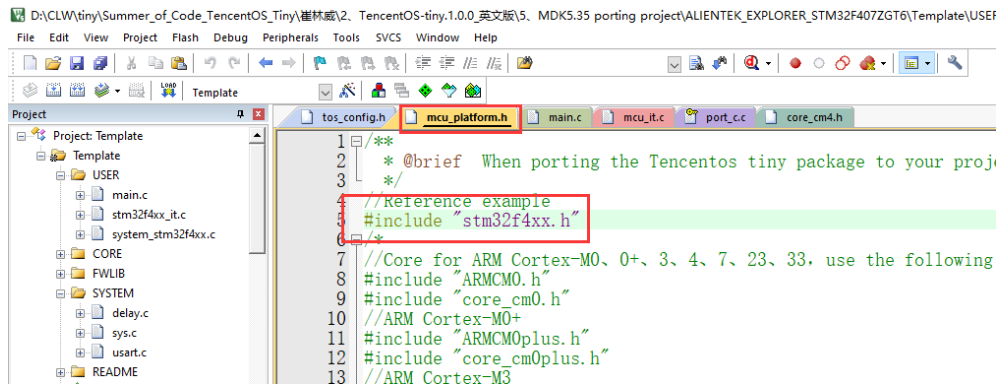


图 3.16 修改 mcu\_platform.h

(4) 接下来，使用如下 main 程序：

```
#include "stm32f4xx.h"
#include "usart.h"
#include "tos_k.h"
k_task_t task1;
k_task_t task2;
k_stack_t task_stack1[1024];
k_stack_t task_stack2[1024];
void test_task1(void *Parameter)
{
    while(1)
    {
        printf("task1 running\r\n");
        tos_task_delay(200);
    }
}
void test_task2(void *Parameter)
{
    k_err_t err;
    printf("task2 running\r\n");
    tos_task_delay(2000);
    // suspend task1 暂停
    printf("suspend task1\r\n");
    err = tos_task_suspend(&task1);
    if(err != K_ERR_NONE)
        printf("suspend task1 fail! code : %d\r\n",err);
    tos_task_delay(2000);
    // resume task1 恢复
    printf("resume task1\r\n");
    err = tos_task_resume(&task1);
    if(err != K_ERR_NONE)
        printf("resume task1 fail! code : %d\r\n",err);
    tos_task_delay(2000);
    // destroy task1 销毁
    printf("destroy task1\r\n");
```

```

err = tos_task_destroy(&task1);
if(err != K_ERR_NONE)
    printf("destroy task1 fail! code : %d \r\n",err);
// task2 running
while(1)
{
    printf("task2 running\r\n");
    tos_task_delay(1000);
}
}
int main(void)
{
    k_err_t err;
    /*初始化 USART 配置模式为 115200 8-N-1, 中断接收*/
    uart_init(115200);
    printf("Welcome to TencentOS tiny\r\n");
    tos_knl_init(); // TOS Tiny kernel initialize
    tos_robin_default_timeslice_config((k_timeslice_t)500u);
    printf("create task1\r\n");
    err = tos_task_create(&task1, "task1", test_task1, NULL, 3, task_stack1, 1024, 20);
    if(err != K_ERR_NONE)
        printf("TencentOS Create task1 fail! code : %d \r\n",err);
    printf("create task2\r\n");
    err = tos_task_create(&task2, "task2", test_task2, NULL, 4, task_stack2, 1024, 20);
    if(err != K_ERR_NONE)
        printf("TencentOS Create task2 fail! code : %d \r\n",err);
    tos_knl_start(); // Start TOS Tiny
}

```

(5) 点击编译，并利用 ST LINK-V2 将程序下载到单片机上，如图 3.17，然后将单片机的串口与电脑连接起来，利用 XCOM 串口通讯助手进行查看，结果如图 3.18 所示。

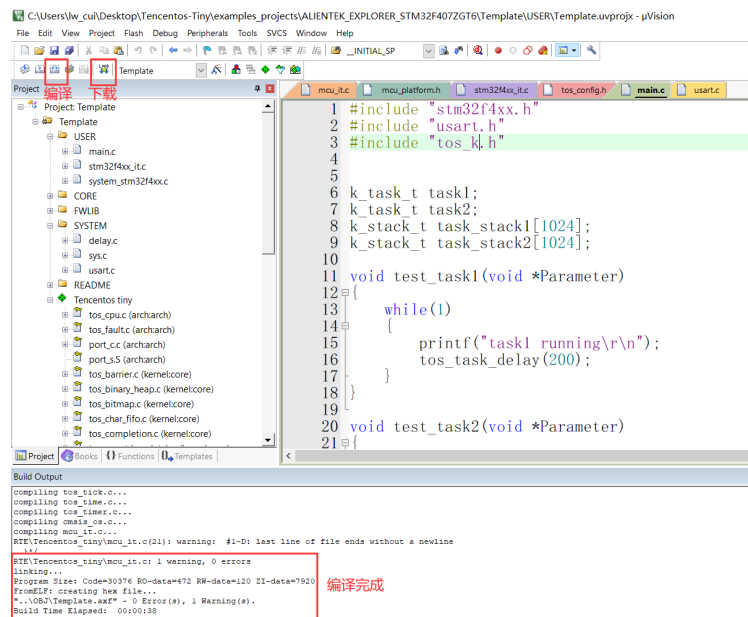
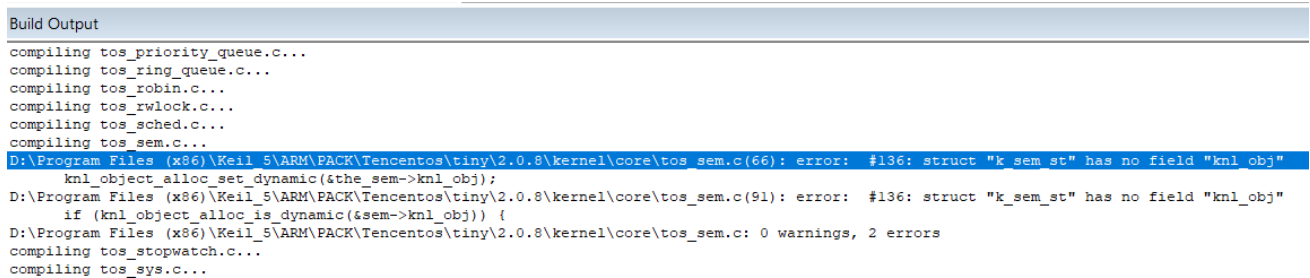


图 3.17 编译界面

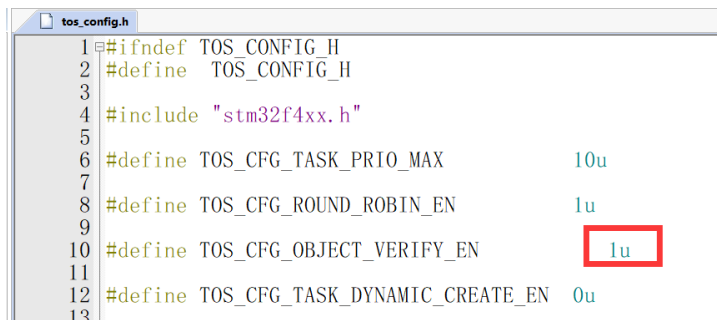


图 3.18 测试界面

另外，在编译过程中如果遇到图 3.19(a)的报错，需要将图 3.19(b)中的#define TOS\_CFG\_OBJECT\_VERIFY\_EN 1u 修改为 TOS\_CFG\_OBJECT\_VERIFY\_EN 0u



(a)



(b)

图 3.19 报错修改

## 4、总结

本文首先研究了基于 MDK 完成第三方软件包封装的开发过程，并编写了软件包制作的步骤，然后结合 TencentOS Tiny 物联网操作系统，对其中的 ARM 内核架构下的文件进行了封装，从而设计了基于 TencentOS Tiny 的软件包。

本软件包能够方便开发者快速地将 TencentOS Tiny 操作系统移植到用户的 ARM 内核单片机上，大大节省了开发移植的时间，同时软件包具有自动适配内核和依赖提示的功能，能够提高移植的效率。



## 5、开发参考

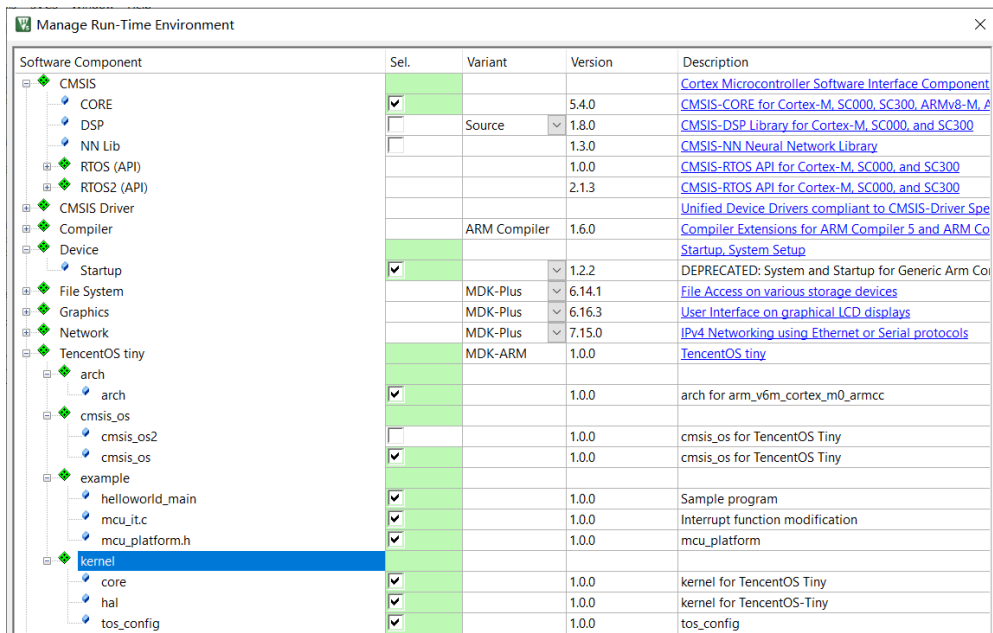
- 1、腾讯物联网操作系统网址 <https://github.com/OpenAtomFoundation/TencentOS-tiny>
  - 2、MDK5 软件包 [MDK5 Software Packs \(keil.com\)](https://www.keil.com/pack/MDK5_Software_Packs/)
  - 3、制作软件包培训视频 <https://www.bilibili.com/video/BV1AK411p7d9>
  - 4、制作软件包博客 [https://blog.csdn.net/qq\\_40259429/article/details/119320319](https://blog.csdn.net/qq_40259429/article/details/119320319)
  - 5、制作简易软件包 <https://www.cnblogs.com/libra13179/p/6273415.html>
  - 6、CMSIS-Driver 软件包 [ARM-software/CMSIS-Driver: Repository of microcontroller peripheral driver implementing the CMSIS-Driver API specification \(github.com\)](https://github.com/ARM-software/CMSIS-Driver)
-

## 6、附录-移植配置参考

### 6.1 MDK5.14 版本移植到 ARM 内核

#### 6.1.1 Cortex-M0 内核移植

(1) Manage Run-Time Environment 勾选如下：



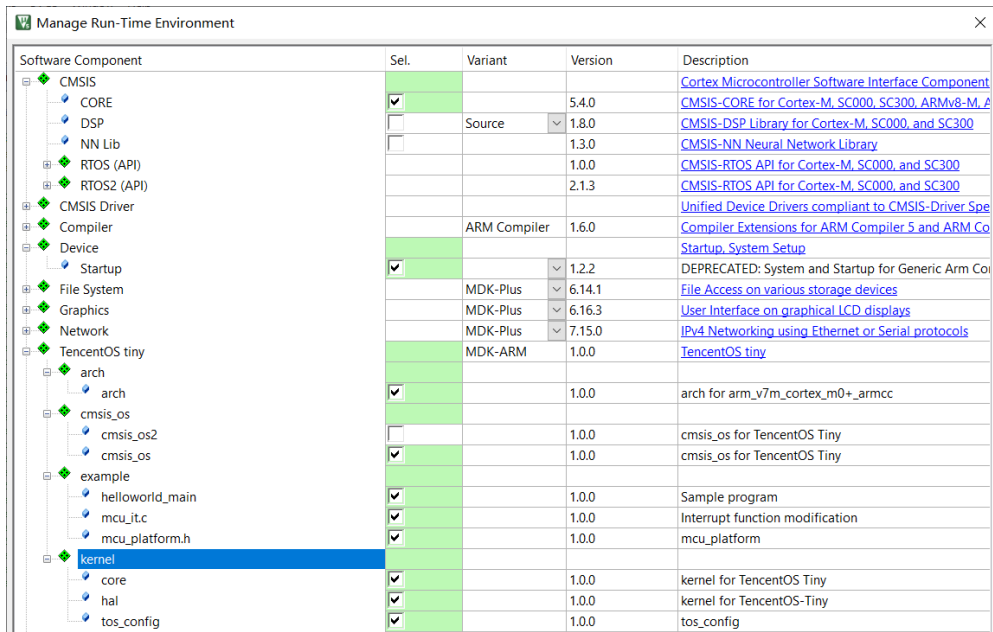
Software Component	Sel.	Variant	Version	Description
CMSIS	<input checked="" type="checkbox"/>			<a href="#">Cortex Microcontroller Software Interface Component</a>
CORE	<input checked="" type="checkbox"/>		5.4.0	<a href="#">CMSIS-CORE for Cortex-M, SC000, SC300, ARMv8-M, and SC300</a>
DSP	<input type="checkbox"/>	Source	1.8.0	<a href="#">CMSIS-DSP Library for Cortex-M, SC000, and SC300</a>
NN Lib	<input type="checkbox"/>		1.3.0	<a href="#">CMSIS-NN Neural Network Library</a>
RTOS (API)	<input type="checkbox"/>		1.0.0	<a href="#">CMSIS-RTOS API for Cortex-M, SC000, and SC300</a>
RTOS2 (API)	<input type="checkbox"/>		2.1.3	<a href="#">CMSIS-RTOS API for Cortex-M, SC000, and SC300</a>
CMSIS Driver	<input type="checkbox"/>			<a href="#">Unified Device Drivers compliant to CMSIS-Driver Specification</a>
Compiler	<input type="checkbox"/>	ARM Compiler	1.6.0	<a href="#">Compiler Extensions for ARM Compiler 5 and ARM Compiler 6</a>
Device	<input type="checkbox"/>			<a href="#">Startup, System Setup</a>
Startup	<input checked="" type="checkbox"/>		1.2.2	DEPRECATED: System and Startup for Generic Arm Cortex-M
File System	<input type="checkbox"/>	MDK-Plus	6.14.1	<a href="#">File Access on various storage devices</a>
Graphics	<input type="checkbox"/>	MDK-Plus	6.16.3	<a href="#">User Interface on graphical LCD displays</a>
Network	<input type="checkbox"/>	MDK-Plus	7.15.0	<a href="#">IPv4 Networking using Ethernet or Serial protocols</a>
TencentOS tiny	<input type="checkbox"/>	MDK-ARM	1.0.0	<a href="#">TencentOS tiny</a>
arch	<input type="checkbox"/>			
arch	<input checked="" type="checkbox"/>		1.0.0	arch for arm_v6m_cortex_m0_armcc
cmsis_os	<input type="checkbox"/>		1.0.0	cmsis_os for TencentOS Tiny
cmsis_os2	<input type="checkbox"/>		1.0.0	cmsis_os for TencentOS Tiny
cmsis_os	<input checked="" type="checkbox"/>		1.0.0	cmsis_os for TencentOS Tiny
example	<input type="checkbox"/>			
helloworld_main	<input checked="" type="checkbox"/>		1.0.0	Sample program
mcu_it.c	<input checked="" type="checkbox"/>		1.0.0	Interrupt function modification
mcu_platform.h	<input checked="" type="checkbox"/>		1.0.0	mcu_platform
kernel	<input checked="" type="checkbox"/>			
core	<input checked="" type="checkbox"/>		1.0.0	kernel for TencentOS Tiny
hal	<input checked="" type="checkbox"/>		1.0.0	kernel for TencentOS-Tiny
tos_config	<input checked="" type="checkbox"/>		1.0.0	tos_config

(2) 在 mcu\_platform.h 中添加：

```
#include "ARMCM0.h"
#include "core_cm0.h"
```

#### 6.1.2 Cortex-M0+内核移植

(1) Manage Run-Time Environment 勾选如下：



Software Component	Sel.	Variant	Version	Description
CMSIS	<input checked="" type="checkbox"/>			<a href="#">Cortex Microcontroller Software Interface Component</a>
CORE	<input checked="" type="checkbox"/>		5.4.0	<a href="#">CMSIS-CORE for Cortex-M, SC000, SC300, ARMv8-M, and SC300</a>
DSP	<input type="checkbox"/>	Source	1.8.0	<a href="#">CMSIS-DSP Library for Cortex-M, SC000, and SC300</a>
NN Lib	<input type="checkbox"/>		1.3.0	<a href="#">CMSIS-NN Neural Network Library</a>
RTOS (API)	<input type="checkbox"/>		1.0.0	<a href="#">CMSIS-RTOS API for Cortex-M, SC000, and SC300</a>
RTOS2 (API)	<input type="checkbox"/>		2.1.3	<a href="#">CMSIS-RTOS API for Cortex-M, SC000, and SC300</a>
CMSIS Driver	<input type="checkbox"/>			<a href="#">Unified Device Drivers compliant to CMSIS-Driver Specification</a>
Compiler	<input type="checkbox"/>	ARM Compiler	1.6.0	<a href="#">Compiler Extensions for ARM Compiler 5 and ARM Compiler 6</a>
Device	<input type="checkbox"/>			<a href="#">Startup, System Setup</a>
Startup	<input checked="" type="checkbox"/>		1.2.2	DEPRECATED: System and Startup for Generic Arm Cortex-M
File System	<input type="checkbox"/>	MDK-Plus	6.14.1	<a href="#">File Access on various storage devices</a>
Graphics	<input type="checkbox"/>	MDK-Plus	6.16.3	<a href="#">User Interface on graphical LCD displays</a>
Network	<input type="checkbox"/>	MDK-Plus	7.15.0	<a href="#">IPv4 Networking using Ethernet or Serial protocols</a>
TencentOS tiny	<input type="checkbox"/>	MDK-ARM	1.0.0	<a href="#">TencentOS tiny</a>
arch	<input type="checkbox"/>			
arch	<input checked="" type="checkbox"/>		1.0.0	arch for arm_v7m_cortex_m0+_armcc
cmsis_os	<input type="checkbox"/>		1.0.0	cmsis_os for TencentOS Tiny
cmsis_os2	<input type="checkbox"/>		1.0.0	cmsis_os for TencentOS Tiny
cmsis_os	<input checked="" type="checkbox"/>		1.0.0	cmsis_os for TencentOS Tiny
example	<input type="checkbox"/>			
helloworld_main	<input checked="" type="checkbox"/>		1.0.0	Sample program
mcu_it.c	<input checked="" type="checkbox"/>		1.0.0	Interrupt function modification
mcu_platform.h	<input checked="" type="checkbox"/>		1.0.0	mcu_platform
kernel	<input checked="" type="checkbox"/>			
core	<input checked="" type="checkbox"/>		1.0.0	kernel for TencentOS Tiny
hal	<input checked="" type="checkbox"/>		1.0.0	kernel for TencentOS-Tiny
tos_config	<input checked="" type="checkbox"/>		1.0.0	tos_config

(2) 在 mcu\_platform.h 中添加：

```
#include "ARMCM0plus.h"
#include "core_cm0plus.h"
```

6.1.3 Cortex-M3 内核移植

(1) Manage Run-Time Environment 勾选如下：

Manage Run-Time Environment				
Software Component	Sel.	Variant	Version	Description
CMSIS	<input checked="" type="checkbox"/>			Cortex Microcontroller Software Interface Component
CORE	<input checked="" type="checkbox"/>		5.4.0	CMSIS-CORE for Cortex-M, SC000, SC300, ARMv8-M, A
DSP	<input type="checkbox"/>	Source	1.8.0	CMSIS-DSP Library for Cortex-M, SC000, and SC300
NN Lib	<input type="checkbox"/>		1.3.0	CMSIS-NN Neural Network Library
RTOS (API)	<input type="checkbox"/>		1.0.0	CMSIS-RTOS API for Cortex-M, SC000, and SC300
RTOS2 (API)	<input type="checkbox"/>		2.1.3	CMSIS-RTOS API for Cortex-M, SC000, and SC300
CMSIS Driver	<input type="checkbox"/>			Unified Device Drivers compliant to CMSIS-Driver Spe
Compiler	<input type="checkbox"/>	ARM Compiler	1.6.0	Compiler Extensions for ARM Compiler 5 and ARM Co
Device	<input type="checkbox"/>			Startup, System Setup
Startup	<input checked="" type="checkbox"/>		1.2.2	DEPRECATED: System and Startup for Generic Arm Co
File System	<input type="checkbox"/>	MDK-Plus	6.14.1	File Access on various storage devices
Graphics	<input type="checkbox"/>	MDK-Plus	6.16.3	User Interface on graphical LCD displays
Network	<input type="checkbox"/>	MDK-Plus	7.15.0	IPv4 Networking using Ethernet or Serial protocols
TencentOS tiny	<input type="checkbox"/>	MDK-ARM	1.0.0	TencentOS tiny
arch	<input type="checkbox"/>			
arch	<input checked="" type="checkbox"/>		1.0.0	arch for arm_v7m_cortex_m3_armcc
cmsis_os	<input type="checkbox"/>		1.0.0	cmsis_os for TencentOS Tiny
cmsis_os2	<input type="checkbox"/>		1.0.0	cmsis_os for TencentOS Tiny
cmsis_os	<input checked="" type="checkbox"/>		1.0.0	cmsis_os for TencentOS Tiny
example	<input type="checkbox"/>			
helloworld_main	<input checked="" type="checkbox"/>		1.0.0	Sample program
mcu_it.c	<input checked="" type="checkbox"/>		1.0.0	Interrupt function modification
mcu_platform.h	<input checked="" type="checkbox"/>		1.0.0	mcu_platform
kernel	<input type="checkbox"/>			
core	<input checked="" type="checkbox"/>		1.0.0	kernel for TencentOS Tiny
hal	<input checked="" type="checkbox"/>		1.0.0	kernel for TencentOS-Tiny
tos_config	<input checked="" type="checkbox"/>		1.0.0	tos_config

(2) 在 mcu\_platform.h 中添加：

```
#include "ARMCM3.h"
#include "core_cm3.h"
```

6.1.4 Cortex-M4 内核移植

(1) Manage Run-Time Environment 勾选如下：

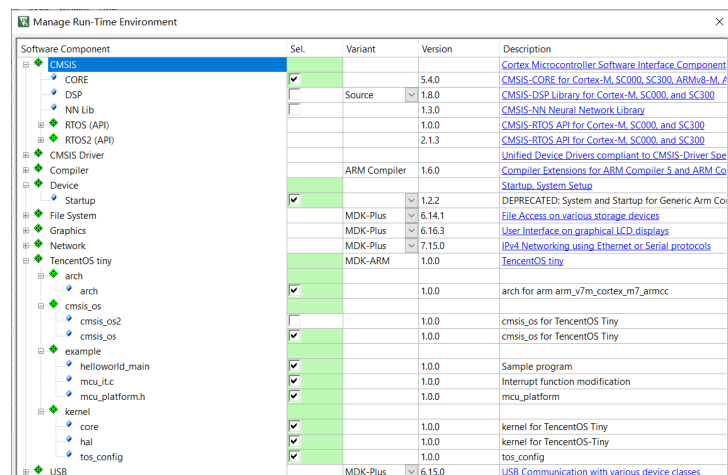
Manage Run-Time Environment				
Software Component	Sel.	Variant	Version	Description
CMSIS	<input checked="" type="checkbox"/>			Cortex Microcontroller Software Interface Component
CORE	<input checked="" type="checkbox"/>		5.4.0	CMSIS-CORE for Cortex-M, SC000, SC300, ARMv8-M, A
DSP	<input type="checkbox"/>	Source	1.8.0	CMSIS-DSP Library for Cortex-M, SC000, and SC300
NN Lib	<input type="checkbox"/>		1.3.0	CMSIS-NN Neural Network Library
RTOS (API)	<input type="checkbox"/>		1.0.0	CMSIS-RTOS API for Cortex-M, SC000, and SC300
RTOS2 (API)	<input type="checkbox"/>		2.1.3	CMSIS-RTOS API for Cortex-M, SC000, and SC300
CMSIS Driver	<input type="checkbox"/>			Unified Device Drivers compliant to CMSIS-Driver Spe
Compiler	<input type="checkbox"/>	ARM Compiler	1.6.0	Compiler Extensions for ARM Compiler 5 and ARM Co
Device	<input type="checkbox"/>			Startup, System Setup
Startup	<input checked="" type="checkbox"/>		1.2.2	DEPRECATED: System and Startup for Generic Arm Co
File System	<input type="checkbox"/>	MDK-Plus	6.14.1	File Access on various storage devices
Graphics	<input type="checkbox"/>	MDK-Plus	6.16.3	User Interface on graphical LCD displays
Network	<input type="checkbox"/>	MDK-Plus	7.15.0	IPv4 Networking using Ethernet or Serial protocols
TencentOS tiny	<input type="checkbox"/>	MDK-ARM	1.0.0	TencentOS tiny
arch	<input type="checkbox"/>			
arch	<input checked="" type="checkbox"/>		1.0.0	arch for arm_v7m_cortex_m4_armcc
cmsis_os	<input type="checkbox"/>		1.0.0	cmsis_os for TencentOS Tiny
cmsis_os2	<input type="checkbox"/>		1.0.0	cmsis_os for TencentOS Tiny
cmsis_os	<input checked="" type="checkbox"/>		1.0.0	cmsis_os for TencentOS Tiny
example	<input type="checkbox"/>			
helloworld_main	<input checked="" type="checkbox"/>		1.0.0	Sample program
mcu_it.c	<input checked="" type="checkbox"/>		1.0.0	Interrupt function modification
mcu_platform.h	<input checked="" type="checkbox"/>		1.0.0	mcu_platform
kernel	<input type="checkbox"/>			
core	<input checked="" type="checkbox"/>		1.0.0	kernel for TencentOS Tiny
hal	<input checked="" type="checkbox"/>		1.0.0	kernel for TencentOS-Tiny
tos_config	<input checked="" type="checkbox"/>		1.0.0	tos_config

(2) 在 mcu\_platform.h 中添加：

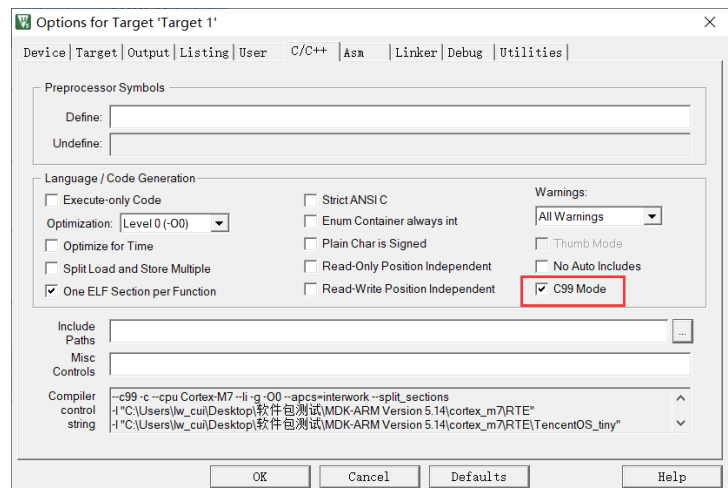
```
#include "ARMCM4.h"
#include "core_cm4.h"
```

### 6.1.5 Cortex-M7 内核移植

1、Manage Run-Time Environment 勾选如下：



## 2、在 MDK 中修改为 C99



### 3、在 mcu\_platform.h 中添加:

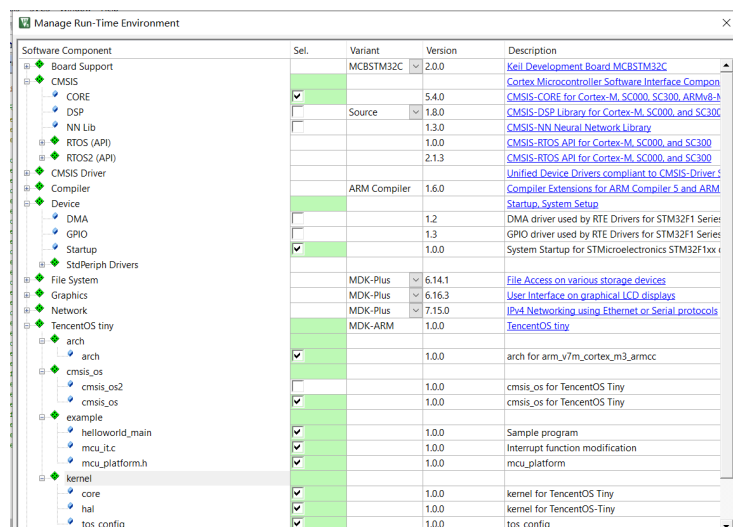
```
#include "ARMCM7.h"
```

```
#include "core_cm7.h"
```

## 6.2 MDK5.14 版本移植到基于 ARM 内核的芯片

### 6.2.1 移植到 stm32f103c8 芯片

1、Manage Run-Time Environment 勾选如下：

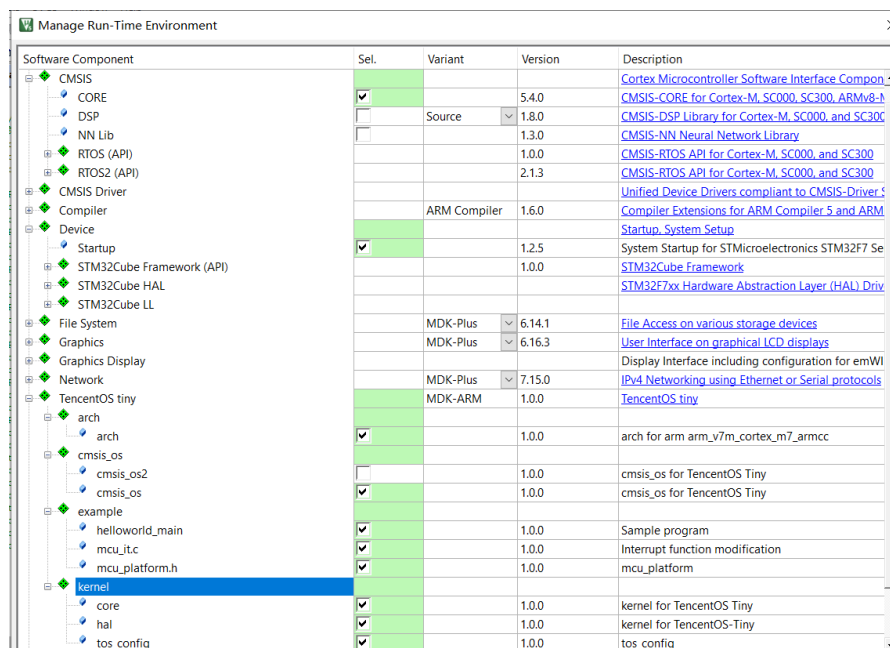


2、在 mcu\_platform.h 中添加：

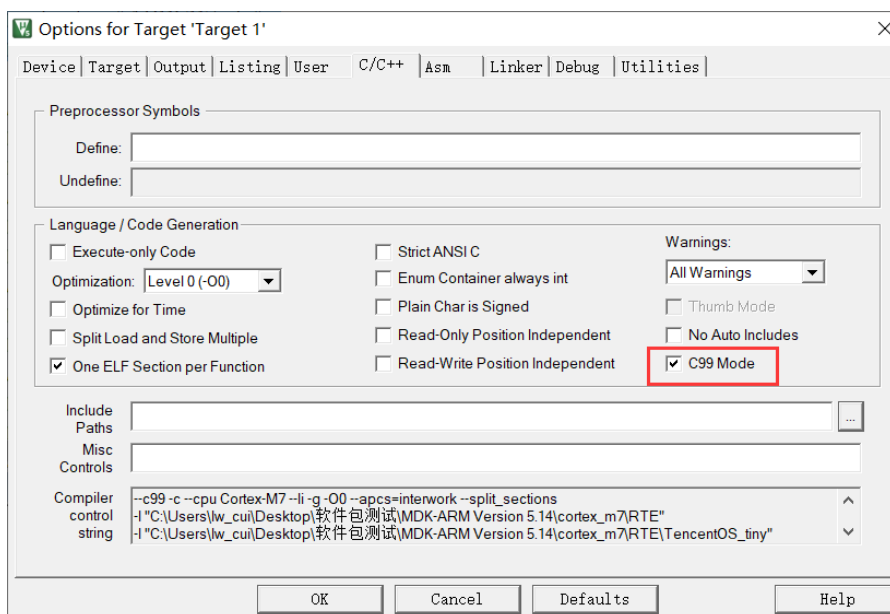
```
#include "stm32f10x.h"
#include "core_cm3.h"
#include "system_stm32f10x.h"
```

### 6.2.2 移植到 stm32f767igt 芯片

1、Manage Run-Time Environment 勾选如下：



2、在 MDK 中修改为 C99

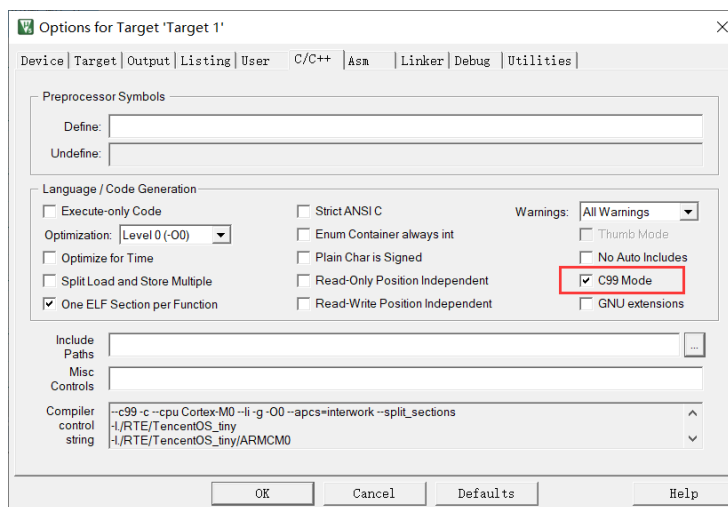
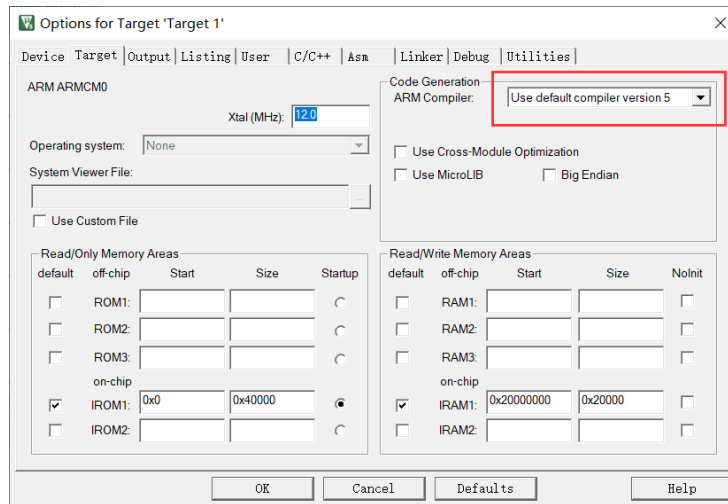


3、在 mcu\_platform.h 中添加：

```
#include "stm32f7xx.h"
#include "core_cm7.h"
#include "system_stm32f7xx.h"
```

### 6.3 MDK5.30 和 MDK5.35 版本移植（Cortex-M0+、0、3、4、7 内核和芯片）

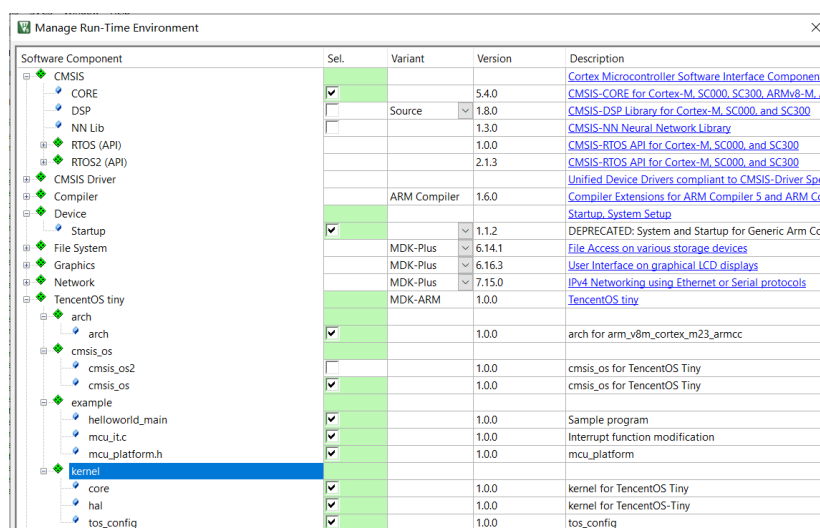
在 MDK5.30 和 MDK5.35 版本的 Keil 移植 TencentOS-tiny Pack 时，对于 Cortex-M0+、0、3、4、7 的内核和芯片，勾选组件和添加头文件的步骤之前一致，但是需要修改编译器的版本，具体修改如下，然后执行编译即可。



## 6.4 MDK5.30 和 MDK5.35 版本移植（Cortex-M23、33）

### 6.4.1 Cortex-M23 内核移植

(1) Manage Run-Time Environment 勾选如下：

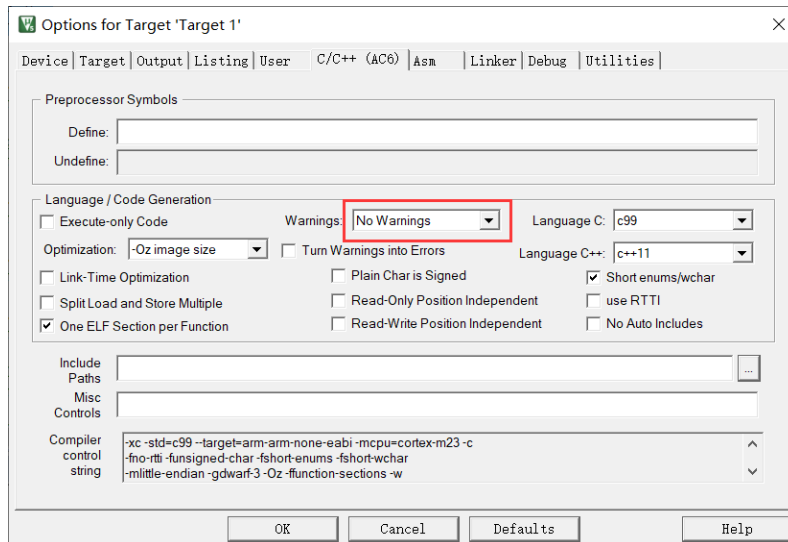


(2) 在 mcu\_platform.h 中添加：

```
#include "ARMCM23.h"
```

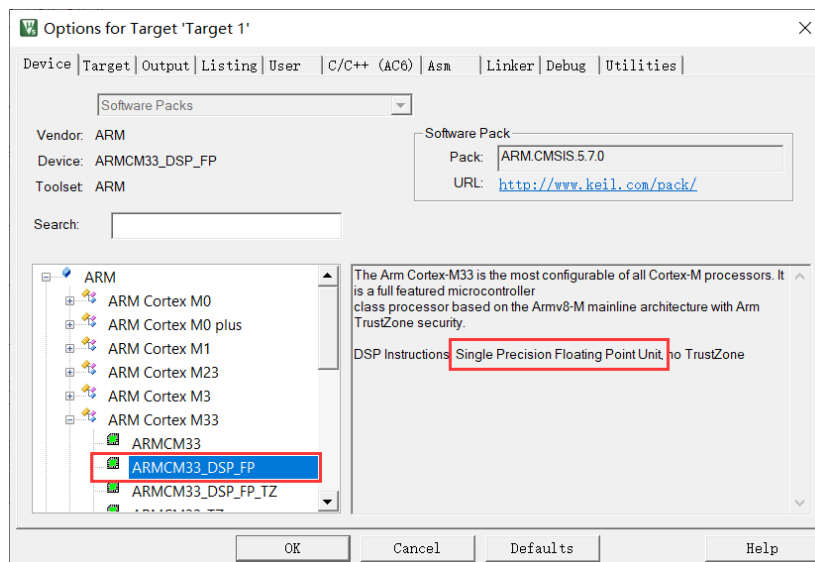
```
#include "core_cm23.h"
```

(3) 修改为不查看报错：



## 6.4.2 Cortex-M33 内核移植

### (1) 选择带 FPU 的芯片



### (2) Manage Run-Time Environment 勾选如下:

Software Component	Sel.	Variant	Version	Description
CMSIS	<input checked="" type="checkbox"/>		5.4.0	Cortex Microcontroller Software Interface Component
CORE	<input checked="" type="checkbox"/>			CMSIS-CORE for Cortex-M, SC000, SC300, ARMv8-M, A
DSP	<input type="checkbox"/>	Source	1.8.0	CMSIS-DSP Library for Cortex-M, SC000, and SC300
NN Lib	<input type="checkbox"/>		1.3.0	CMSIS-NN Neural Network Library
RTOS (API)	<input type="checkbox"/>		1.0.0	CMSIS-RTOS API for Cortex-M, SC000, and SC300
RTOS2 (API)	<input type="checkbox"/>		2.1.3	CMSIS-RTOS API for Cortex-M, SC000, and SC300
CMSIS Driver	<input type="checkbox"/>			Unified Device Drivers compliant to CMSIS-Driver Spe
Compiler	<input type="checkbox"/>	ARM Compiler	1.6.0	Compiler Extensions for ARM Compiler 5 and ARM Co
Device	<input type="checkbox"/>			Startup, System Setup
Startup	<input checked="" type="checkbox"/>		1.2.2	DEPRECATED: System and Startup for Generic Arm Co
File System	<input type="checkbox"/>	MDK-Plus	6.14.1	File Access on various storage devices
Graphics	<input type="checkbox"/>	MDK-Plus	6.16.3	User Interface on graphical LCD displays
Network	<input type="checkbox"/>	MDK-Plus	7.15.0	IPv4 Networking using Ethernet or Serial protocols
TencentOS tiny	<input type="checkbox"/>	MDK-ARM	1.0.0	TencentOS tiny
arch	<input checked="" type="checkbox"/>		1.0.0	arch for arm_v8m_cortex_m33_armcc
cmsis_os	<input type="checkbox"/>		1.0.0	cmsis_os for TencentOS Tiny
cmsis_os2	<input type="checkbox"/>		1.0.0	cmsis_os for TencentOS Tiny
example	<input type="checkbox"/>			
helloworld_main	<input checked="" type="checkbox"/>		1.0.0	Sample program
mcu_itc	<input checked="" type="checkbox"/>		1.0.0	Interrupt function modification
mcu_platform.h	<input checked="" type="checkbox"/>		1.0.0	mcu_platform
kernel	<input checked="" type="checkbox"/>			
core	<input checked="" type="checkbox"/>		1.0.0	kernel for TencentOS Tiny
hal	<input checked="" type="checkbox"/>		1.0.0	kernel for TencentOS-Tiny
tos_config	<input checked="" type="checkbox"/>		1.0.0	tos_config



(3) 在 mcu\_platform.h 中添加:

```
#include "ARMCM33_DSP_FP.h"
```

```
#include "core_cm33.h"
```

(4) 修改为不查看报错:

