RTOS

《实时操作系统应用技术—基于RT-Thread与ARM的编程实践》教学脉络

苏州大学 王宜怀，yihuaiw@suda.edu.cn

2024年5月18日

内容简介

嵌入式实时操作系统是嵌入式人工智能与物联网终端的重要工具。本书以国产RT-Thread实时操作系统为蓝本，以ARM架构MCU为载体，基于应用开发视角，阐述实时操作系统的线程、调度、延时函数、事件、消息队列、信号量、互斥量等基本知识要素，重点给出实时操作系统下程序设计方法。对与原理部分，仅从知其然，也希望了解其所以然的角度，用一章篇幅以在内核代码中注入显示输出方式给出原理浅析。全书分为9章，分别为基本概念与线程基础知识、第一个样例工程、 应用程序的基本要素、同步与通信的应用方法、底层硬件驱动构件、程序设计方法、嵌入式人工智能、物联网应用、初步理解RT-Thread的调度原理等。

目 录

一、实时操作系统的教学目的

几个层面：应用开发、了解原理、理解原理、撰写RTOS

学习RTOS的几个可能出发点：

（1）应用开发：学会在RTOS场景下进行基本应用程序开发；（绝大多数目标）

（2）了解原理：在掌握应用编程的前提下，了解其运行原理，进行深度应用程序开发。（服务于应用开发）

（3）理解原理：知其然，知其所以然；（人数很少：移植与驻留）

（4）撰写一个RTOS。（极少数人做的事情）

进一步说，应用与理解RTOS可以简单地表述为：

应用就是在基本了解线程、调度等基本概念基础上，正确运用RTOS下的延时函数、事件、消息队列、信号量、互斥量等基本要素，进行应用程序的开发，让RTOS成为嵌入式软件开发的辅助工具。（这是教学重点）

理解就是要理解RTOS工作的基本原理，也就就是理解调度机制，理解延时函数、事件、消息队列、信号量、互斥量等基本要素工作机制，为应用编程提供更加坚实的基础。（一般在研究生阶段进行）

以消息队列为例，应用编程就是这样一个场景：有个“装消息”的篮子，有个发送消息处，当发送消息处发出消息，消息就会进入“装消息”的篮子，而只要消息“篮子”中有消息，等待消息的程序就会运行。理解运行机制就是为什么发送消息处发出消息，消息就会自动进入消息“篮子”，为什么消息“篮子”中有消息，等待消息的程序就会运行。

理解在大多数情况下，很难达到，本书改为了解原理，服务于应用。

二、实时操作系统的教学准备

2.1 实时操作系统的选型问题

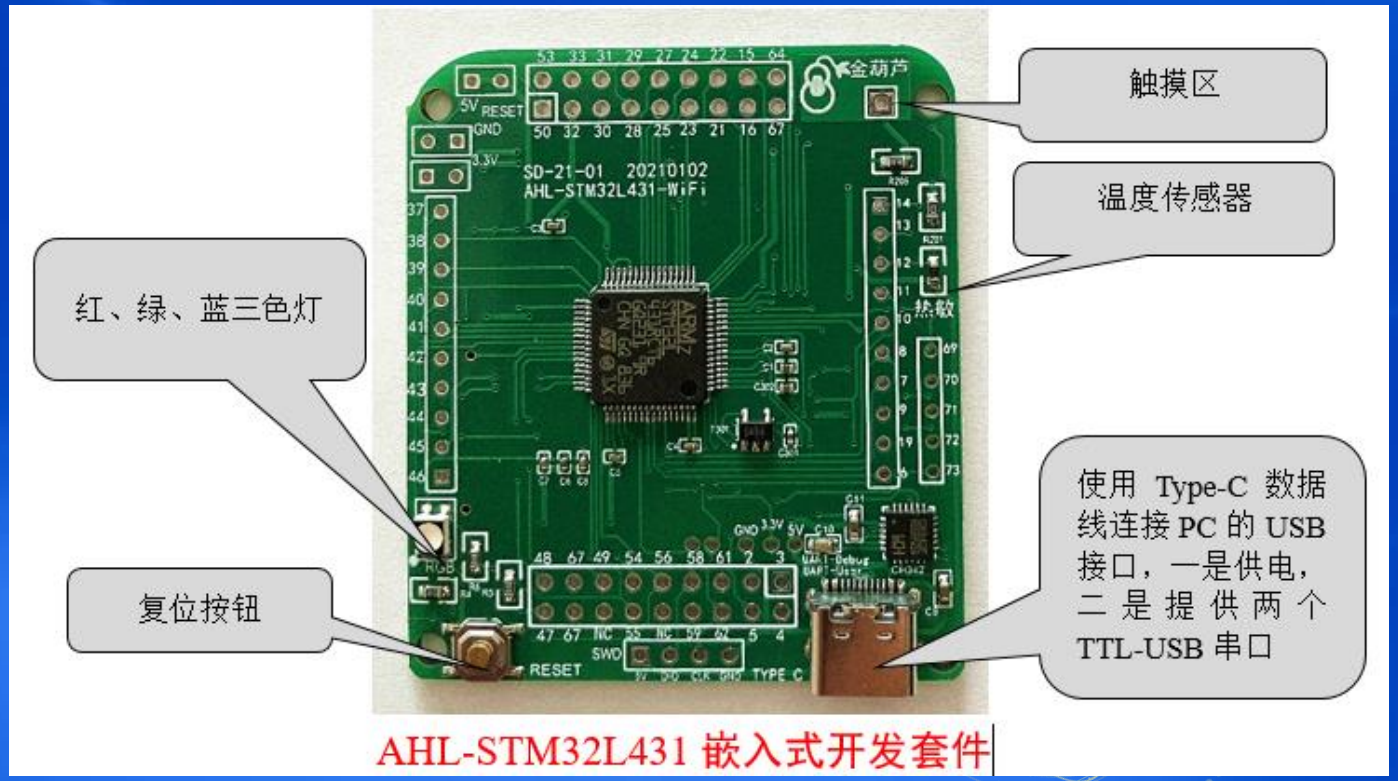
RTOS种类繁多，有国外的，也有国产的；有收费的，也有免费的；有带有持续维护升级的，也有依赖爱好者更新升级的。初学者有时不知所措。但是无论如何，学习RTOS，必须以一个具体的RTOS为蓝本。不同RTOS，其应用方法及原理大同小异，掌握其共性是学习的关键，这样才能达到举一反三的效果。这里给出的国产RT-Thread实时操作系统是上海睿赛德电子科技有限公司于2006年开始推出的开源RTOS，面向嵌入式人工智能与物联网领域，被广泛应用于工业控制等众多行业领域，将其内核用到实际产品中，没有收费陷阱问题。

2.2 实践硬件

RTOS是理论联系实践的典型体系，学习RTOS，不仅要以一个具体的RTOS为蓝本，还要一个具体的MCU为蓝本进行实践，否则就可能成为纸上谈兵。

对实践器材的要求是：满足RTOS的基本知识要素实践训练，价格低廉，使用方便，稳定可靠。

（本书赠送）



2.3 开发环境

开发环境：苏州大学嵌入式人工智能与物联网实验经过十多年努力研发了AHL-GEC-IDE（http://sumcu.suda.edu.cn/AHLwGECwIDE/main.htm），可以方便地用于RTOS实践。针对AHL-STM32L431，兼容STM32CubeIDE。

2.4 设计实践样例

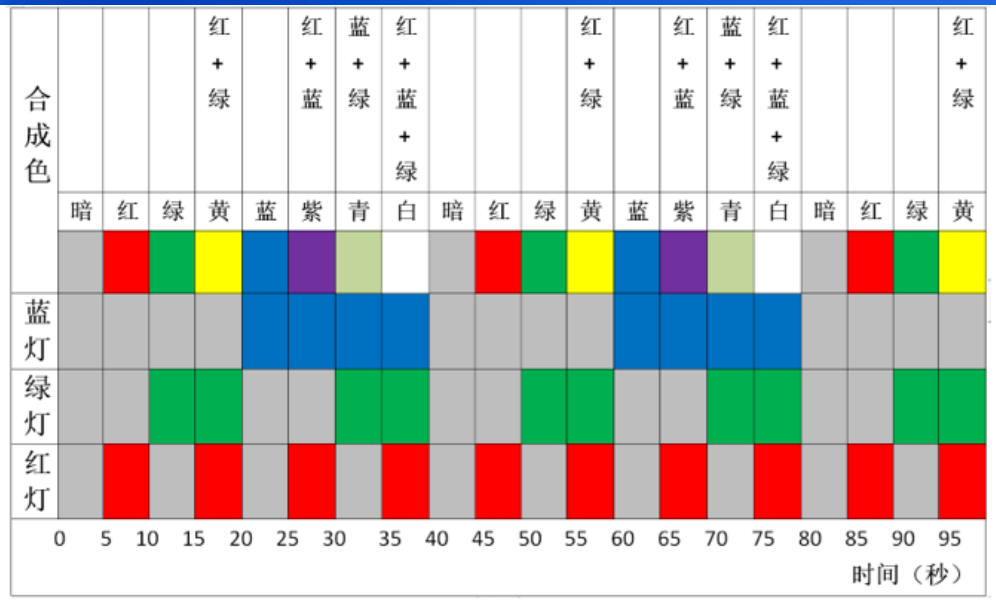
“照葫芦画瓢”教学实践基本理念：照葫芦画瓢是比喻照着样子模仿，表示简单易行之含义。古希腊哲学家亚里士多德说过：“人从儿童时期起就有模仿本能，他们用模仿而获得了最初的知识，模仿就是学习”。孟子则曰：“大匠诲人必以规矩，学者亦必以规矩”。借此，期望通过建立符合软件工程基本原理的“葫芦”，为“依葫芦画瓢”提供坚实基础，达到降低学习实践难度之目标。

建立标准工程框架：做到“分门别类，各有归处”

01_Doc	文档文件夹：文档作为工程密切相关部分，是软件工程的基本要求
02_CPU	CPU 文件夹：存放 CPU 相关文件
03_MCU	MCU 文件夹：含有 linker_file、startup、MCU_drivers 下级文件夹
04_GEC	GEC 文件夹：引入通用嵌入式计算机（GEC）概念，预留该文件夹
05_UserBoard	用户板文件夹：含有硬件接线信息的 User.h 文件及应用驱动
06_SoftComponent	软件构件文件夹：含有与硬件无关的软件构件
07_AppPrg	应用程序文件夹：应用程序主要在此处编程

工程框架

第一个样例功能



第一个样例程序功能

第一个样例工程中的线程

归属	线程名	执行函数	优先级	线程功能	中文含义
内核	main_thread	app_init	10	创建其他线程	主线程
	idle	idle	31	空闲线程	空闲线程
用户	thd_redlight	thread_redlight	10	红灯以 5s 为周期闪烁	红灯线程
	thd_greenlight	thread_greenlight	10	绿灯以 10s 为周期闪烁	绿灯线程
	thd_bluelight	hread_bluelight	10	蓝灯以 20s 为周期闪烁	蓝灯线程

2.5 教材的撰写

第1篇 基础应用篇

第2篇 综合实践篇

第3篇 原理剖析篇

2.6 为RTOS教学所做的准备工作总结

（1）**实践硬件**：AHL-STM32L431

（2）**开发环境**：AHL-GEC-IDE
(https://sumcu.suda.edu.cn/AHLwGECwIDE/list.htm)

（3）**样例程序编制**：针对各基本要素，按照统一模板编制了样例程序
(https://sumcu.suda.edu.cn/RTwThreadwARMjc/list.htm)

（4）**撰写书籍**：嵌入式实时操作系统--基于RT-Thread的EAI&IoT系统开发，机械工业出版社，2021年；**实时操作系统应用技术--基于RT-Thread与ARM的编程实践，机械工业出版社，2024年（本次交流）**

（5）**课件**：实时操作系统应用技术-课件V1.0-20240502.rar
(https://sumcu.suda.edu.cn/RTwThreadwARMjc/list.htm)

（6）**科研拓展**：嵌入式人工智能（第7章）、基于WiFi物联网（第8章）

三、RTOS中核心概念导引

3.1 线程

线程是RTOS中最重要的概念之一。在RTOS下，把一个复杂的嵌入式应用工程按一定规则分解成一个个功能清晰的小工程，然后设定各个小工程的运行规则，交给RTOS管理，这就是基于RTOS编程的基本思想。这一个个小工程被称为线程（Thread），RTOS管理这些线程，被称为调度（Scheduling）。

补充说明：多进程与多线程

RTOS一般是单进程的，所以不再出现进程概念，这是RTOS由RTOS应用场合决定的，用于连续运行的实时性要求较高的工业场景，而台式计算机、笔记本电脑、平板电脑、手机等是多进程的，如看似同时打开WORD、播放音乐、发邮件等，每个正在运行的程序均是一个进程，每个进程一般会有若干线程

要给RTOS中的线程下一个准确而完整的定义并不十分容易，可以从不同视角理解线程。从线程调度视角理解，可以认为，RTOS中的线程是一个功能清晰的小程序，是RTOS调度的基本单元；从RTOS的软件设计视角来理解，就是在软件设计时，需要根据具体应用，划分出独立的、相互作用的程序集合，这样的程序集合就被称为线程，每个线程都被赋予一定的优先级；从CPU视角理解，在单CPU下，某一时刻CPU只会处理（执行）一个线程，或者说只有一个线程占用CPU。RTOS内核的关键功能就是以合理的方式为系统中的每个线程分配时间（即调度），使之得以运行。

实际上，根据特定的RTOS，线程可能被称为任务（Task），也可能使用其他名词，含义或许稍有差异，但本质不变，也不必花过多精力追究其精确语义，因为学习RTOS的关键在于掌握线程设计方法、理解调度过程、提高编程鲁棒性、理解底层驱动原理、提高程序规范性、可移植性与可复用性、提高嵌入式系统的实际开发能力等。

3.2 调度

调度（Scheduling）就是决定该轮到哪个线程运行了，它是内核最重要的职责。每个线程根据其重要程度不同，被赋予一定的优先级。不同的调度算法对RTOS的性能有较大影响，基于优先级的调度算法是RTOS常用的调度算法，其核心思想是，总是让处于就绪态的、优先级最高的线程先运行。然而何时高优先级线程掌握CPU的使用权，由使用的内核类型确定，基于优先级的内核有不可抢占型和可抢占型两种类型。

3.3 线程的三要素：线程函数、线程堆栈、线程描述符

线程函数。一个线程，对应一段函数代码，完成一定功能。从代码上看，线程函数与一般函数并无区别。

线程堆栈。它是独立于线程函数之外的RAM，按照“后进先出”策略组织的一段连续存储空间，是RTOS中线程概念的重要组成部分。在 RTOS中被创建的每个线程都有自己私有的堆栈空间，在线程的运行过程中，堆栈用于保存线程程序运行过程中的局部变量、线程调用普通函数时会为线程保存返回地址等参数变量、保存线程的上下文等。

线程描述符。线程被创建时，系统会为每个线程创建一个唯一的线程描述符（Task Descriptor，TD），它相当于线程在RTOS中的一个“身份证”，RTOS就是通过这些“身份证”来管理线程和查询线程信息的。

3.4 线程的四种状态：终止态、阻塞态、就绪态和激活态

RTOS中的线程一般有四种状态，分别为终止态、阻塞态、就绪态和激活态。在任一时刻，线程被创建后所处的状态一定是四种状态之一。

终止态：线程已经完成或被删除，不再需要使用CPU。

阻塞态：又可称为“挂起态”。线程未准备好，不能被激活，因为该线程需要等待一段时间或某些情况发生；当等待时间到或等待的情况发生时，该线程才变为就绪态，处于阻塞态的线程描述符存放于等待列表或延时列表中。

就绪态：线程已经准备好可以被激活，但未进入激活态，因为其优先级等于或低于当前的激活线程，一旦获取CPU的使用权就可以进入激活态，处于就绪态的线程描述符存放于就绪列表中。

激活态：又称“运行态”，该线程在运行中，线程拥有CPU使用权。

3.5 RTOS下的延时函数

ROTS中，线程一般不采用原地空跑（空循环）的方式进行延时（该方式线程仍然占用CPU的使用权），而往往会使用到延时函数（**该方式线程会让出CPU使用权**），通过使用延时列表管理延时线程，从而实现对线程的延时。在RTOS下使用延时函数，内核把暂时不需执行的线程，插入到延时列表中，让出CPU的使用权，并对线程进行调度。

3.6 事件、消息队列、信号量与互斥量

在RTOS中，当为了协调中断与线程之间或者线程与线程之间同步，但不需要传送数据时，常采用**事件**作为手段。

在RTOS中，如果需要在线程间或线程与中断间传送数据，那就需要采用消息队列作为**同步与通信**手段。

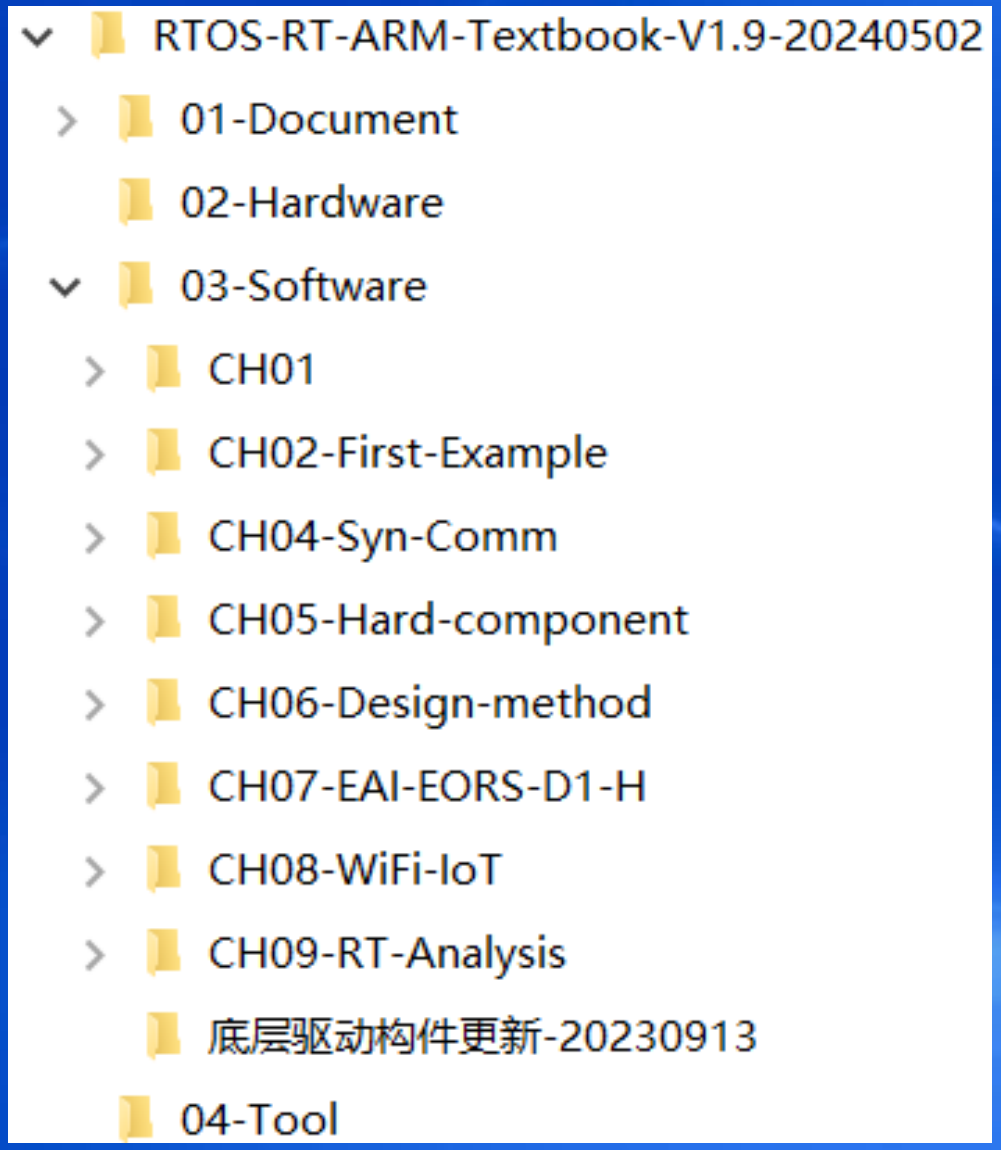
当共享资源有限时，可以采用**信号量**来表达资源可使用的次数，当线程获得信号量时就可以访问该共享资源了。

当共享资源只有一个时，为了确保在某个时刻只有一个线程能够访问该共享资源，可以考虑采用**互斥量**来实现。

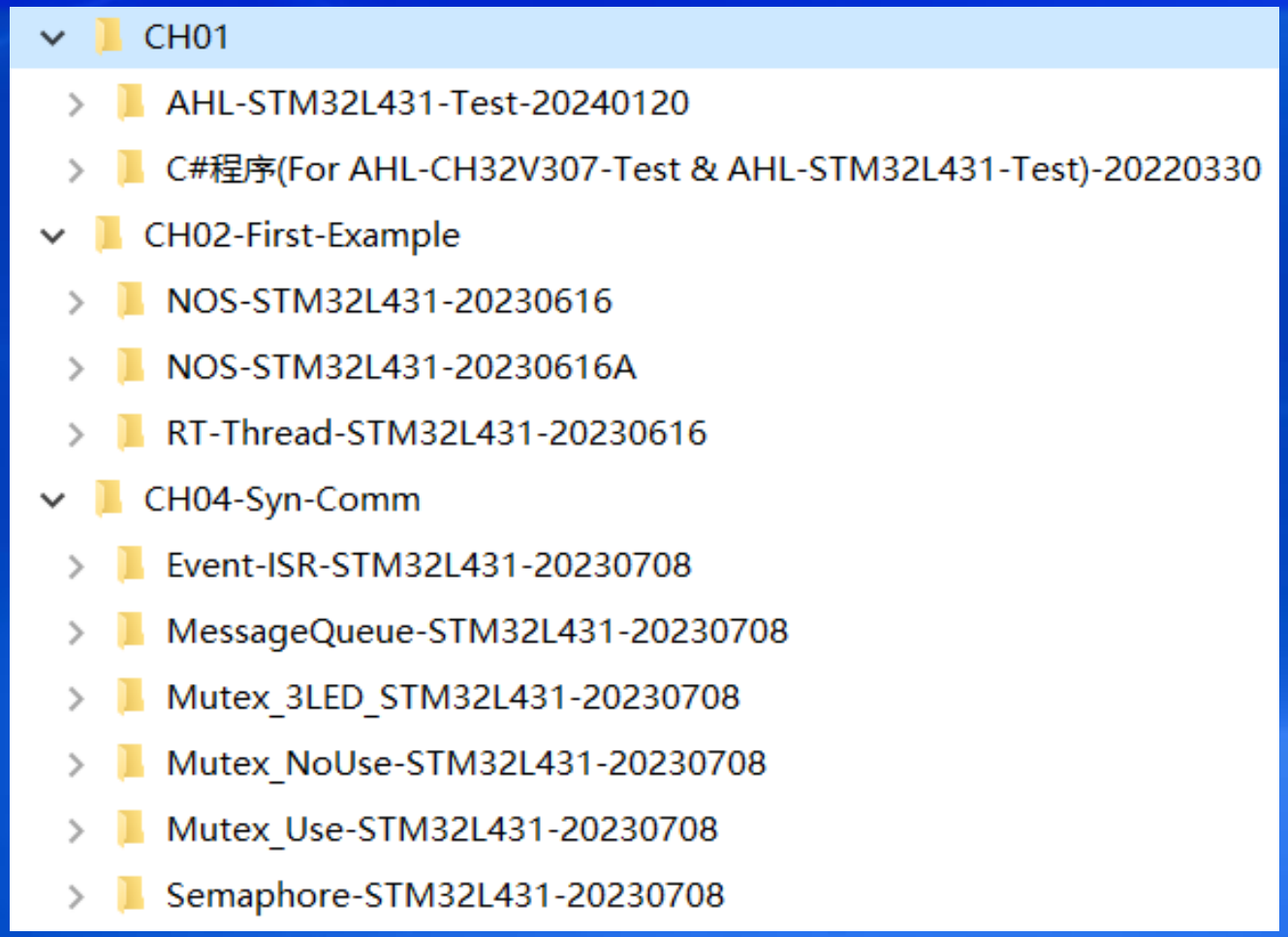
四、实践源代码导引

电子资源下载：

https://sumcu.suda.edu.cn/RTwThreadwARMjc/list.htm


RTOS-RT-ARM-Textbook-V1.9-20240502
01-Document
02-Hardware
03-Software
CH01
CH02-First-Example
CH04-Syn-Comm
CH05-Hard-component
CH06-Design-method
CH07-EAI-EORS-D1-H
CH08-WiFi-IoT
CH09-RT-Analysis
底层驱动构件更新-20230913
04-Tool

五、实践操作演示


CH01
AHL-STM32L431-Test-20240120
C#程序(For AHL-CH32V307-Test & AHL-STM32L431-Test)-20220330
CH02-First-Example
NOS-STM32L431-20230616
NOS-STM32L431-20230616A
RT-Thread-STM32L431-20230616
CH04-Syn-Comm
Event-ISR-STM32L431-20230708
MessageQueue-STM32L431-20230708
Mutex_3LED_STM32L431-20230708
Mutex_NoUse-STM32L431-20230708
Mutex_Use-STM32L431-20230708
Semaphore-STM32L431-20230708


Thank You