RT-Thread

RT-Thread IoT OS
Global Tech Conference 2022

RTduino
RT-Thread的Arduino生态兼容层

满鉴霆 (Meco)

RT-Thread社区核心开发和维护者

meco@rt-thread.org

https://github.com/mysterywolf

https://github.com/RTduino/RTduino

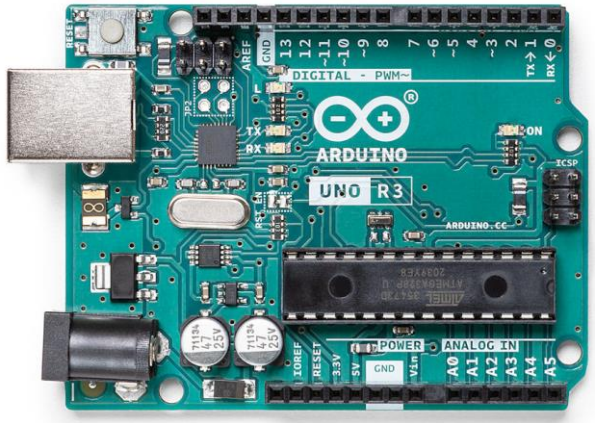
Arduino

- 开发于2005年
- 基于16位AVR单片机
- 让非电子专业的学生可以快速实现一些电子方面想法和设计

包括:

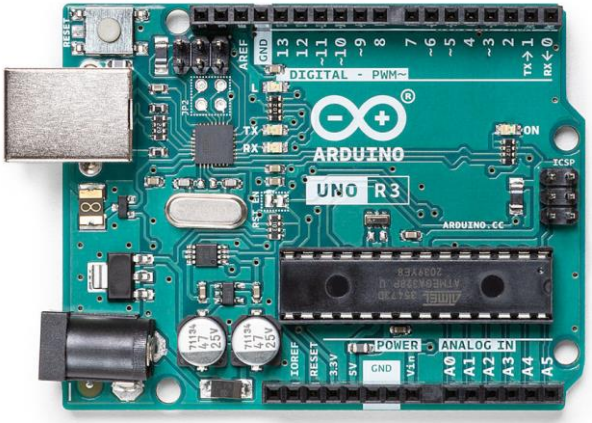
- 集成开发环境IDE
- 标准的硬件开发板(Arduino UNO)
- 方便快捷API和第三方库

生态越来越丰富
- 硬件生态
- 软件生态

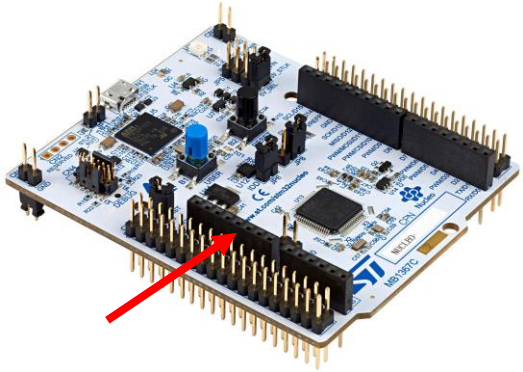


Arduino的硬件生态

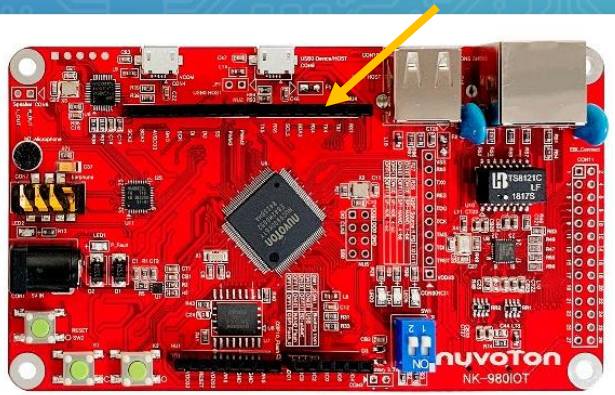
RT-Thread



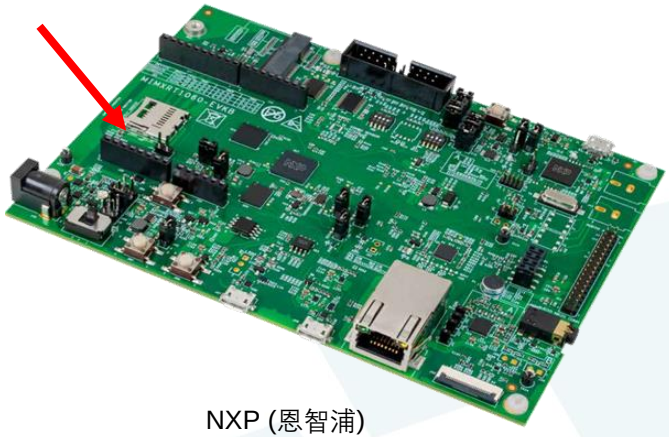
半导体厂商与Arduino硬件生态



意法半导体

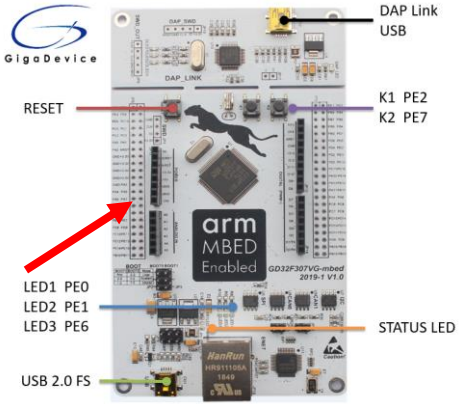


新唐科技

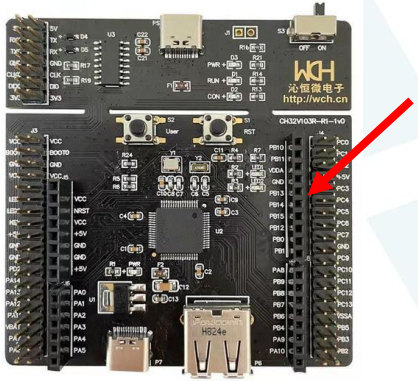


NXP (恩智浦)

中科蓝讯



GD兆易创新



沁恒微电子

.....

Arduino的软件生态

RT-Thread

Libraries

The Arduino environment can be extended through the use of libraries, just lik
Libraries provide extra functionality for use in sketches, e.g. working with harc
library in a sketch, select it from **Sketch > Import Library**.
A number of libraries come installed with the IDE, but you can also download
instructions for details on installing libraries. There is also a tutorial on writing
Guide for information on making a good Arduino-style API for your library.

- Communication (1054)
- Data Processing (271)
- Data Storage (142)
- Device Control (849)
- Display (425)
- Other (398)
- Sensors (966)
- Signal Input/Output (378)
- Timing (198)
- Uncategorized (175)

通信类：1054个
数据处理类：271个
数据存储类：142个
设备控制类：849个
显示类：425个
其他：398个
传感器类：966个
信号输入输出类：378个
时间相关：198个
未归类：175个
共计：4856个

半导体厂商的评估板有一个缺口：只兼容了Arduino硬件生态，但是没有兼容软件生态。

这个缺口，可以由RT-Thread来弥补，原因：

- RT-Thread有众多的BSP
- RT-Thread有统一的设备驱动框架

无论什么型号的BSP，只要支持RT-Thread设备框架，就可以兼容Arduino生态

因此，RT-Thread的Arduino生态兼容层—RTduino应运而生

兼容Arduino的软件生态：可以在RT-Thread上直接运行Arduino的库，实现和Arduino生态的共用

https://github.com/RTduino/RTduino 欢迎Star！

优势

1. 极大地减低了RT-Thread的入门门槛

学习方式由自下而上，转为自上而下

MCU 寄存器
MCU HAL库
RT-Thread 内核
RT-Thread BSP驱动
RT-Thread设备框架
最终实现设计

应用简单快速实现设计
Arduino生态兼容层
RT-Thread设备框架
RT-Thread BSP驱动
RT-Thread 内核
MCU HAL库
MCU 寄存器

- 帮助用户屏蔽掉RT-Thread和MCU底层相关的内容和知识。让用户快速地专注于上层的业务逻辑和应用，不再拘泥于底层的驱动和操作系统的相关知识
- 借助Arduino生态中的图形化开发软件，可以以图形化编程的方式直接驱动RT-Thread的BSP

https://github.com/RTduino/RTduino

优势

2. 让RT-Thread更具平台化

RT-Thread不仅要服务于专业的嵌入式工程师，也要服务于具有第三学科专业素养的工程师和科研人员

也就是让这些工程师和科研人员以极低的时间成本和精力，快速地把嵌入式板卡和RT-Thread使用起来，进而将精力专注于自己的领域。

操作系统诞生的目的不是为了难为人，而是简化流程，让用户专注于自己的事情，进而提高工作和生产效率。

专业性是对RT-Thread自身的考量，低门槛易学是针对用户一侧。二者没有必然联系，也并不冲突。

https://github.com/RTduino/RTduino

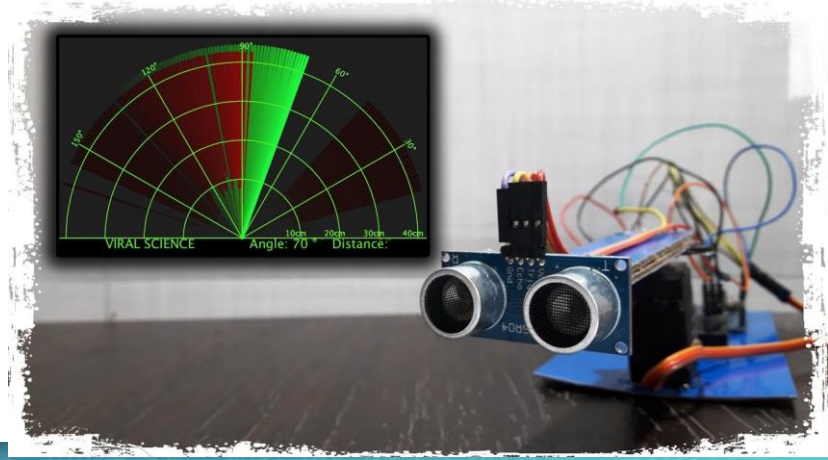
优势

3. 共用Arduino社区生态，极大的丰富了RT-Thread的社区生态

让RT-Thread用户直接使用Arduino的库(例如传感器驱动库、数据处理库等)，通过共用、兼容Arduino的生态，极大的扩充了RT-Thread的生态

- 兼容 ~95% Arduino原生API，只有2个API不计划支持 (详见https://github.com/RTduino/RTduino/issues/2)
- 兼容所有纯软件类Arduino库(例如算法类、Json等)
- 兼容所有I2C传感器类的库
- 兼容单总线类传感器库（如DHT11等）
- 兼容Adafruit传感器框架
- 可以兼容Arduino社区优秀的项目
- 正在推进SPI设备和传感器的兼容
- 不兼容针对Arduino特定板卡（如UNO R3板）的库

https://github.com/RTduino/UltrasoundRadar



https://github.com/RTduino/RTduino

优势（专业嵌入式工程师）

RT-Thread

1. 打通产品原型研发和正样研发的壁垒

- 用Arduino开发原型是真的香！但是把原型转成正样就要自己一点点去重新造轮子了
- 通过RTduino软件包，就无需重新造轮子，可以直接将Arduino库部署在任何一个BSP上

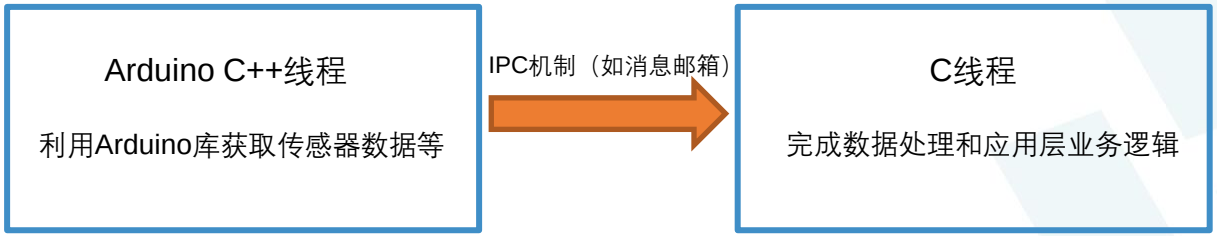
https://github.com/RTduino/RTduino

优势（专业嵌入式工程师）

RT-Thread

2. 借助于RT-Thread线程间通信机制，可以不拘泥于C++编程，实现C++/C混合编程

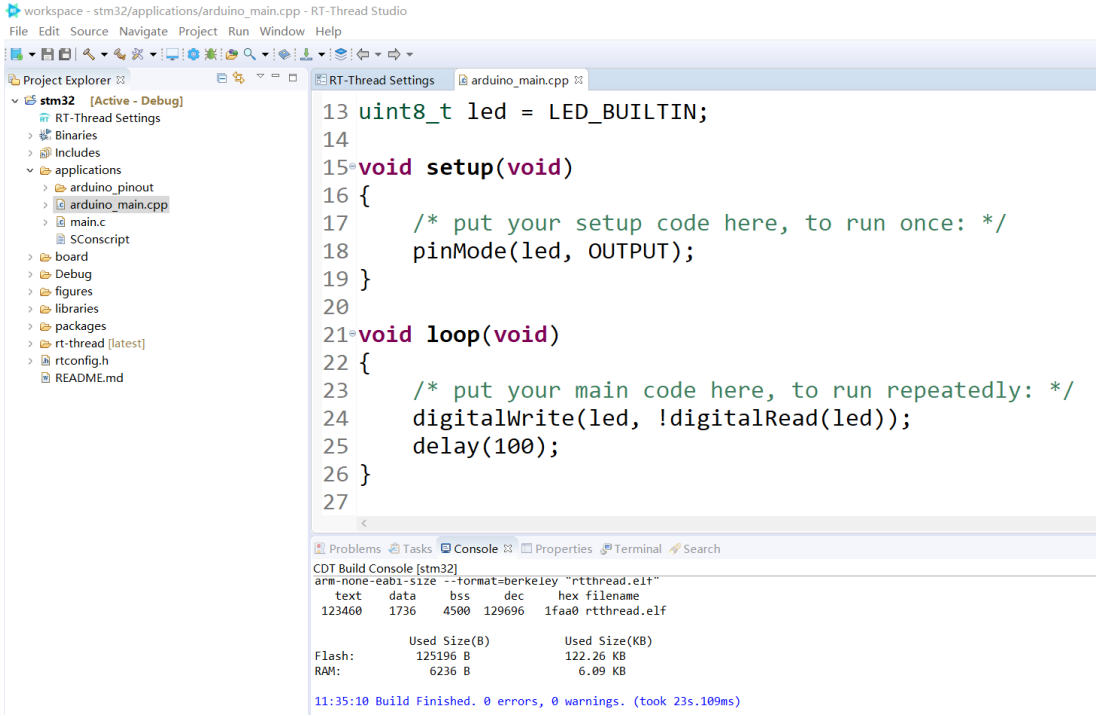
- Arduino C++线程只负责简单的传感器数据获取，通过线程间通信将数据传到C线程当中去处理
- 在发挥Arduino库C++面向对象编程快捷便利的同时，也不需要过多深入了解C++的编程知识
- C和C++各发挥其所长
- 借助于RT-Thread的多线程，不用再拘泥于Arduino的setup – loop编程框架



优势（专业嵌入式工程师）

RT-Thread

3. 支持RT-Thread Studio和Keil-MDK专业嵌入式集成开发环境



摒弃原生Arduino IDE简陋的开发环境

- 无法查找函数和变量的定义位置
- 没有多源码文件管理机制

优势（高校教育）

打通低年级入门，高年级进阶以及毕业工作之间的隔阂。可以做到，所学即所用，产学一体化。

传统：低年级Arduino或8051教学，高年级32位MCU，工作可能还和学校所学不同。

使用RTduino后：

- 低年级借助RTduino来使用RT-Thread，了解嵌入式的基本知识；
- 高年级继续学习RT-Thread内核、设备框架以及MCU相关知识；
- 毕业工作后，会继续用到RT-Thread；
- 始终贯穿着一条主线，在不同阶段以不同的维度来学习和使用RT-Thread和MCU

正在推进RTduino与MATLAB/Simulink的联合仿真的功能

https://www.mathworks.com/hardware-support/arduino-matlab.html

https://github.com/RTduino/RTduino

RT-Thread

Thanks!
演示环节

https://github.com/RTduino/RTduino