



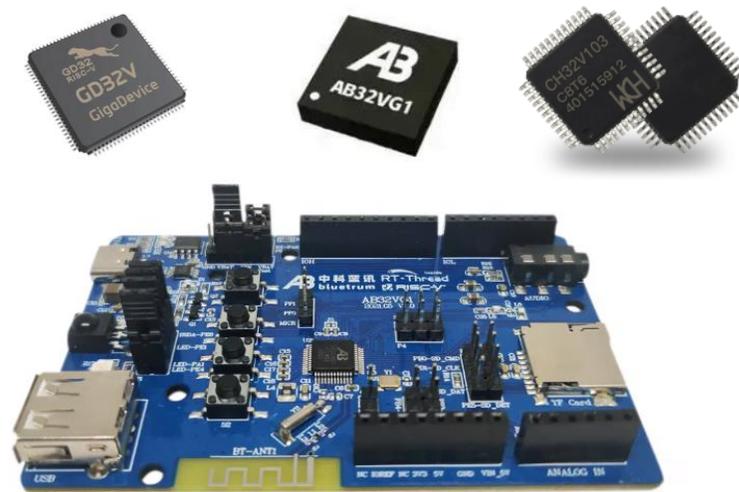
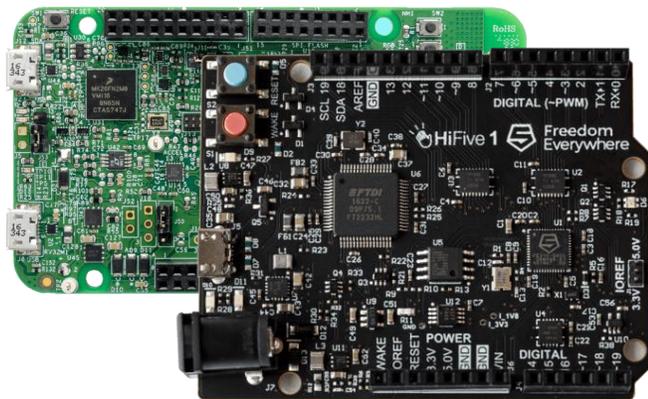
# RT-Thread Smart for RISC-V

熊谱翔 @ [rt-thread.org](http://rt-thread.org)

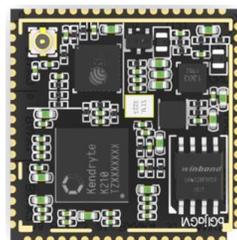
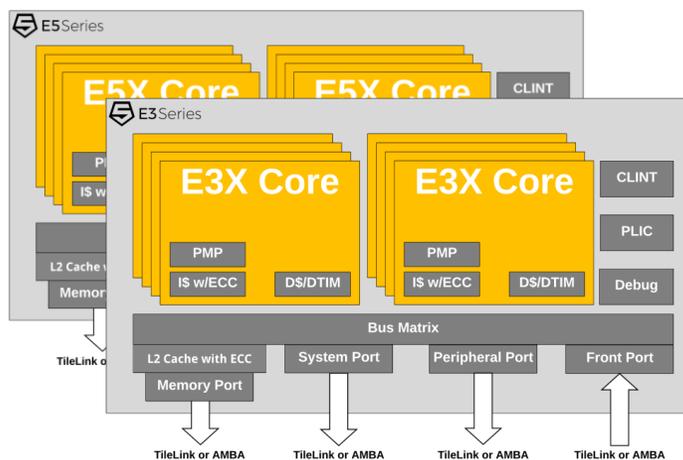
# RT-Thread 对 RISC-V 的支持情况

RT-Thread 自开始就在推进对 RISC-V 架构的支持，同时也包括社区、生态合作伙伴在 RISC-V 方面的贡献。

- MCU
- Emulator: QEMU , JuiceVM



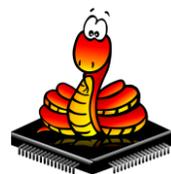
# RT-Thread 对 RISC-V 的支持情况：SMP多核



## 支持SMP对称多核特性

- 整体执行一份操作系统程序
- 任务可以依据CPU负荷情况迁移到空闲核
- 兼容单核RT-Thread API及软件组件、软件包

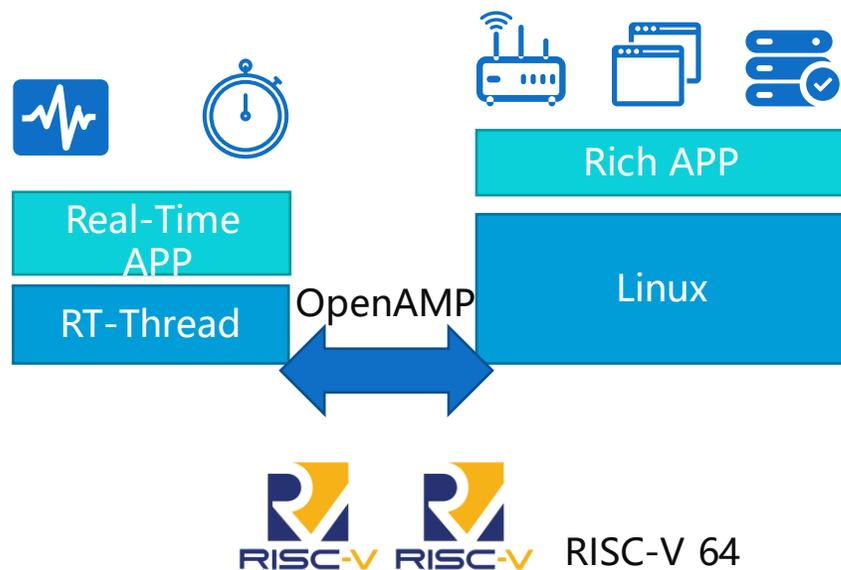
充分利用多核特性提升计算性能，方便多核编程，  
可用于多类应用场合



MicroPython



# RT-Thread对RISC-V的支持情况：AMP多核



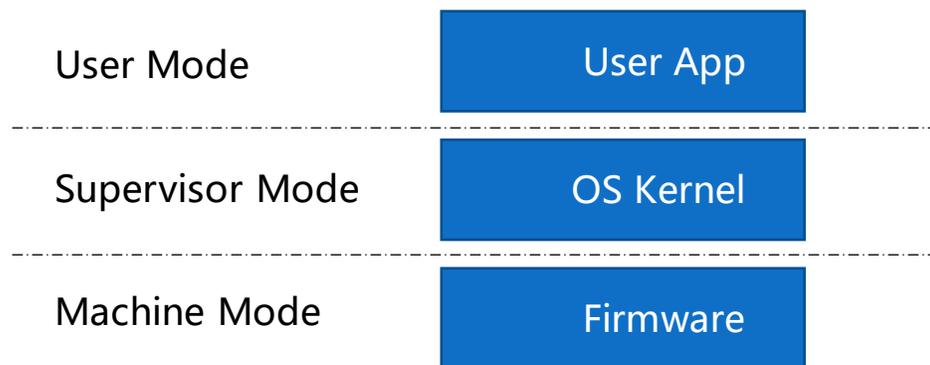
## 在AMP多核上运行异构多系统

- RT-Thread负责实时控制处理
- Linux负责复杂应用
- 底层通过共享内存方式的OpenAMP通信

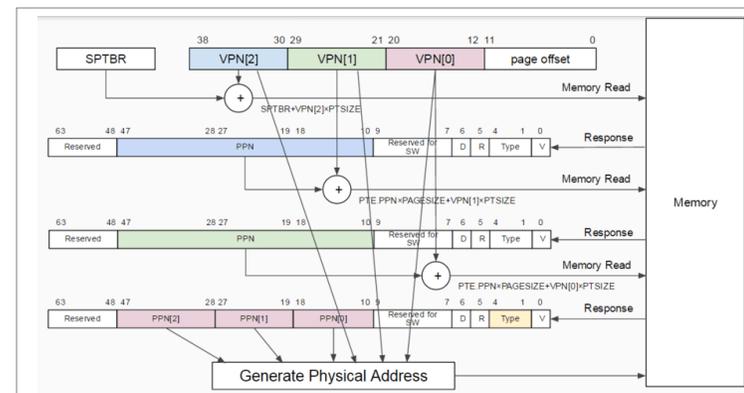


# 从MCU到MPU, 从RTOS到RT-Thread Smart

RISC-V, 从M模式 -> 支持MMU的S/U模式



+ MMU, sv39



传统的RTOS: 运行在线性地址空间, 应用与内核一体化

- 完整基于MMU的进程模型, 混合微内核方式的操作系统。
- RISC-V: 支持K210, qemu-virt64-riscv, 全志D1等处理器



# RT-Thread Smart 混合微内核操作系统

面向微处理器MPU，带MMU，内存保护单元

## 采用微内核架构

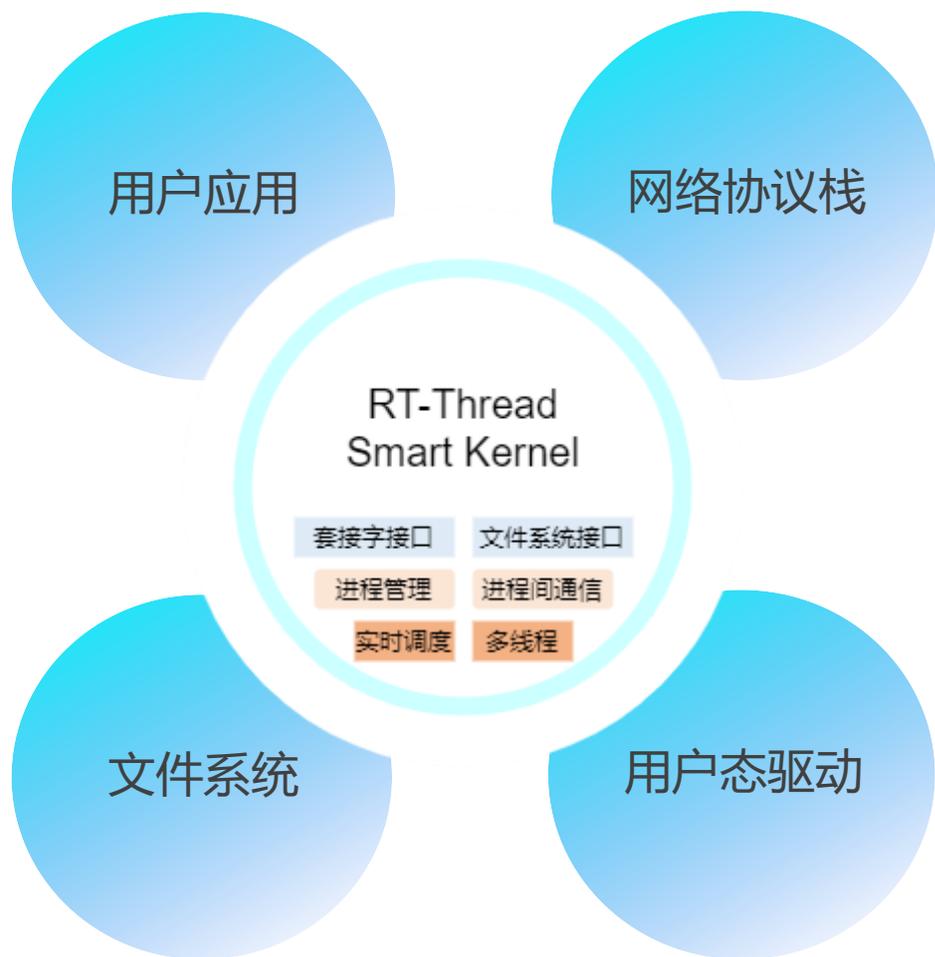
- 更小，更快，更安全
- 可支持单核/多核，32位/64位芯片平台

## 特点

- 使用DRAM的芯片场合，启动速度快，功耗低
- 使能MMU，应用具备独立的地址空间，内存使用更安全
- 相对Linux，资源占用更低，实时性好，安全性更好
- 完整的POSIX环境，Linux程序，C++程序可无缝移植



# RT-Thread Smart 系统架构



## 01. 内核轻型化

- 低至500kB内核尺寸
- 只包含基本功能，同时也可定制（调整系统服务位置）

## 02. 用户态系统服务

- 可拆卸，可重启
- 每个应用进程具备独立的地址空间，相互隔离，安全性好

## 03. 相同的API风格

- 应用与内核都可延续RT-Thread API
- 用户态扩展性强

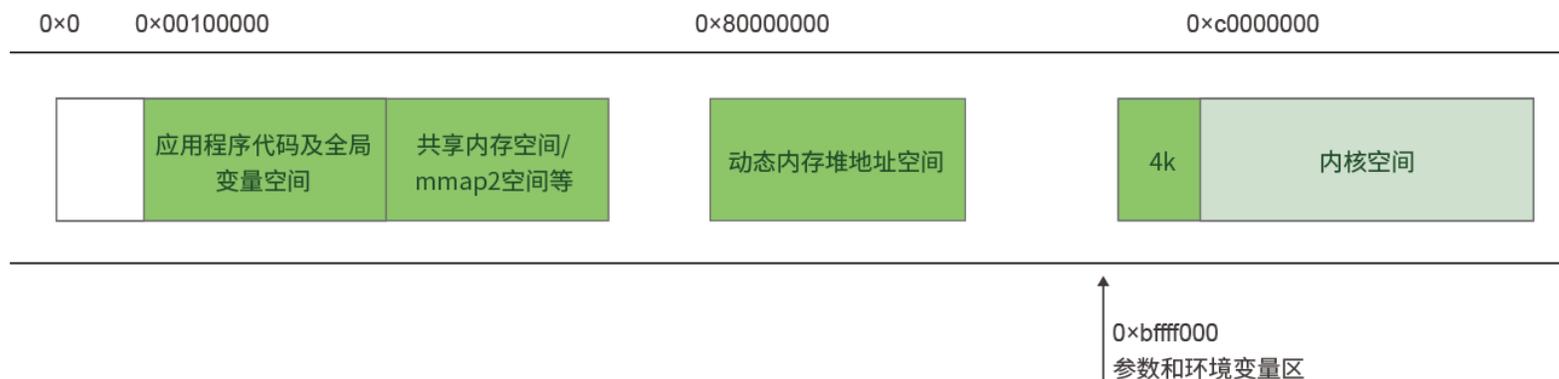


# RT-Thread Smart 更注重功能安全

RT-Thread Smart 分离成:

- 内核态
- 用户态

针对用户态应用，具备全地址空间，由MMU来保证相互之间的地址隔离



# RT-Thread Smart 用户态系统服务

RT-Thread Smart 在用户态可提供常见的系统服务，实现功能和内核的分离：



FSKit

- 用户态的数据存储系统服务
- 支持FAT/exFAT文件系统
- 支持Flash上的文件系统，YAFFS, JFFS2等



NetKit

- 用户态的TCP/IP网络协议栈系统服务
- 完整的BSD socket API接口
- IPv4/6支持



PersimUI Kit

- 前端化的现代UI系统，支持JavaScript小程序开发
- 支持多种常用控件、动画，并可进行自定义，实现各类炫酷效果；支持半透明、Alpha混合、抗锯齿等效果
- 适配嵌入式 GPU，具备 GUI 硬件加速能力

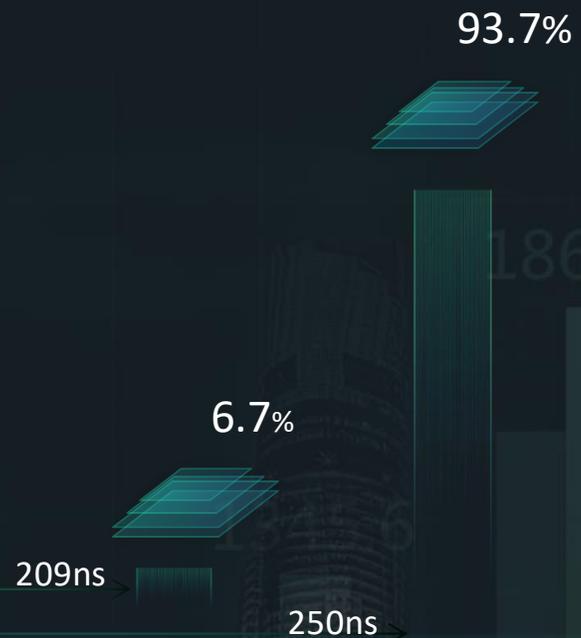


# RT-Thread Smart 延续RT-Thread优异的实时性

RT-Thread Smart具备优异的实时性能

- 中断延时 < 1us

4万个中断响应测试  
占比情况



(在1.2GHz 单核处理器上测试得到)



# RT-Thread Smart 极速启动

可在 **500ms** 以内启动完毕，内核/基本用户态服务进入就绪状态

内核：

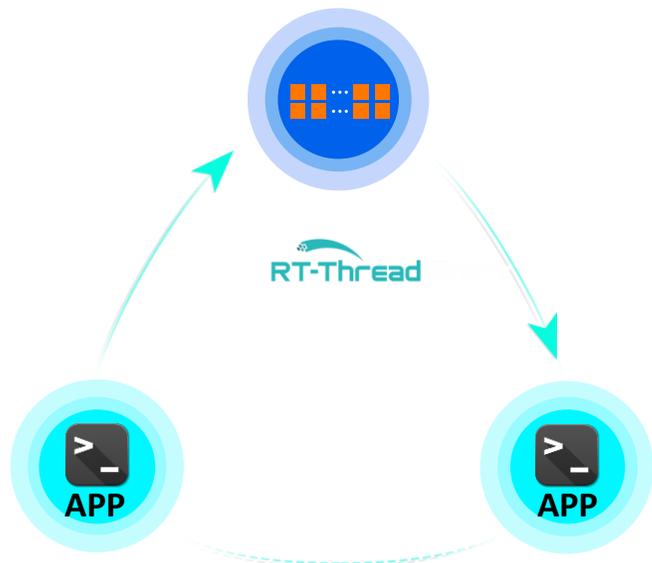
- 内核尺寸可以低至500kB以下
- 可以不依赖于文件系统启动

用户态：

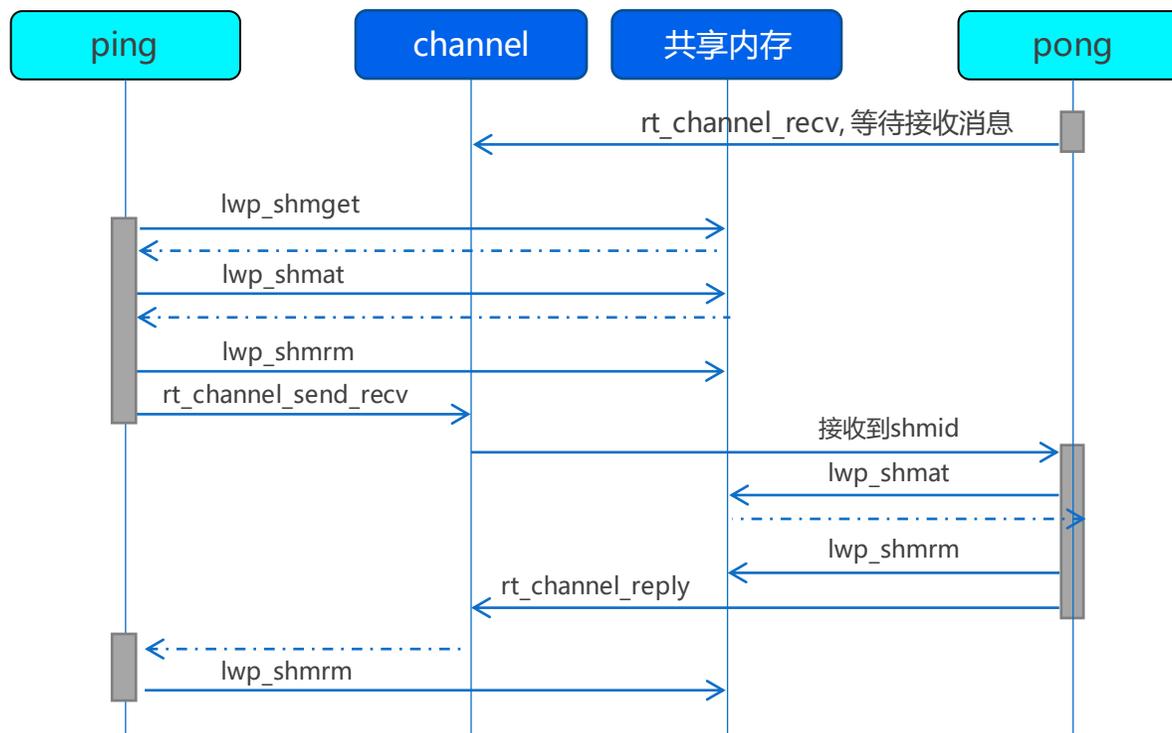
- 用户态支持动态链接方式
- 支持压缩格式的只读文件系统（cromfs）
- 可以做到按需启动服务



# RT-Thread Smart 微内核: 进程间通信



- 进程间以消息方式传递数据，仅消息句柄，速度快、效率高；
- 消息数据放于共享内存空间中
- 以共享内存方式解决进程与进程间数据地址的问题，同时免去数据复制开销



# RT-Thread Smart 完整兼容POSIX标准 API、C++11/14

在用户态完整兼容 PSE53 API标准:

- 支持多进程, 包括经典的fork, vfork, pipe系统调用
- 完整的网络支持, 具备标准的Socket Abstraction Layer (SAL), 并支持UNIX套接字, netlink等
- 支持POSIX signal, threads等



## 01. 完整支持stdc++库

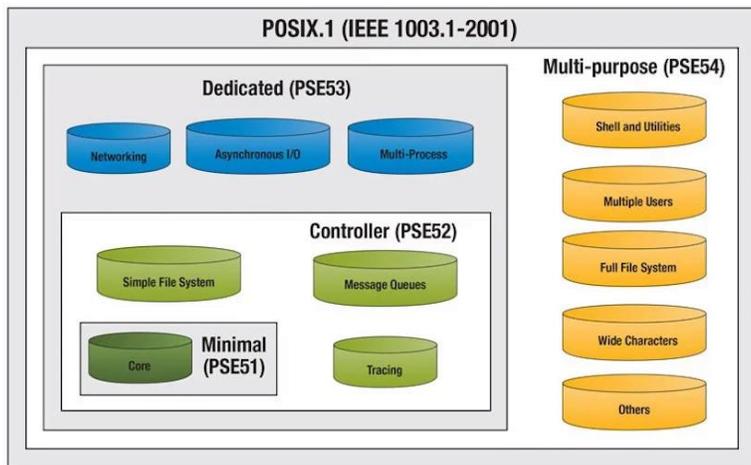
- C++11/14中的标准库, 算法等
- std::vector, std::map, 容器等

## 02. 并发编程及智能指针

- std::thead、std::future
- std::mutex、std::condition\_variable、std::atomic等
- shared\_ptr、unique\_ptr等

## 03. 系列语法糖

- auto自动类型
- lambda表达式及函数等



# RT-Thread Smart GNU Apps生态

可使用GNU Apps传统的编译方式进行编译：

- Makefile
- configure & make
- CMake

广泛可使用的GNU Apps生态：

- openssl, libz, SDL, librws
- wget, curl
- lua, quickjs, sqlite
- uhttpd, lighttpd
- busybox, dropbear ssh

**OpenSSL**  
Cryptography and SSL/TLS Toolkit

 SQLite

 curl

 Lua

 **SDL**  
Simple Directmedia Layer

**BUSYBOX**  




# RT-Thread Smart 兼容已有RT-Thread生态

## 软件包分类



## 01. 用户态RT-Thread API

- 在用户态依然可以兼容RT-Thread应用;
- `rt_thread_create/startup`, `rt_mutex_*`, `rt_malloc/free` etc;

## 02. 兼容软件包脚本

- 依然可以通过menuconfig方式使用软件包;
- 兼容软件包的Kconfig, Sconscript脚本;

## 03. RT-Thread软件包生态

- 多达数百的RT-Thread软件包生态直接使用;
- `webnet/webclient`, `mqtt` etc.



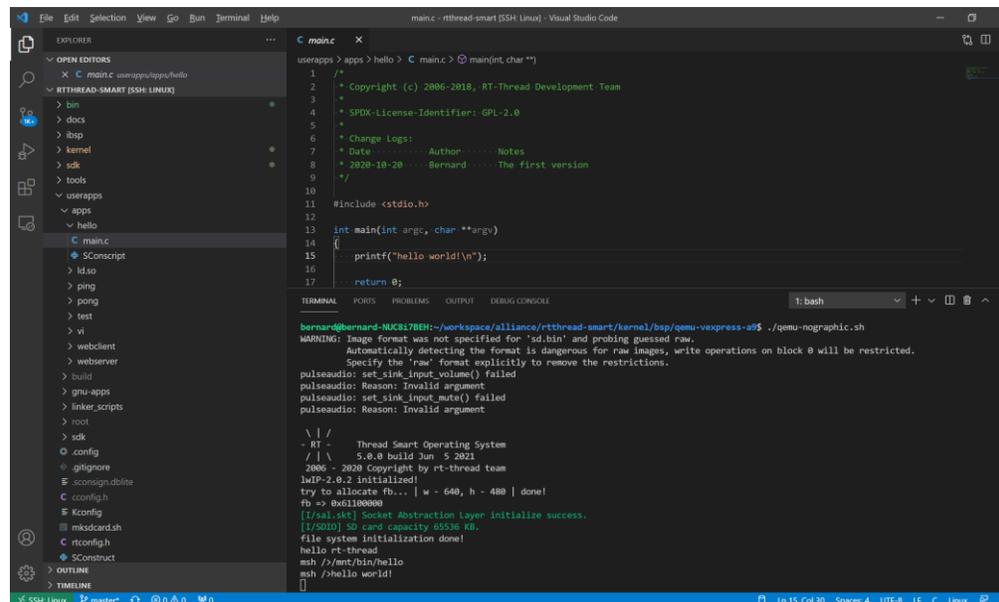
# RT-Thread Smart 开发环境

支持 **vscode** 模式进行开发:

- 创建应用程序
- 连接到远程ssh server进行开发
- 支持直接调试

支持Windows & Linux平台, 推荐使用 **Linux** 平台

- Linux平台: 方便进行移植、调试GNU Apps



```
1 /*
2  * Copyright (c) 2006-2018, RT-Thread Development Team
3  *
4  * SPDX-License-Identifier: GPL-2.0
5  *
6  * Change Logs:
7  * Date           Author       Notes
8  * 2020-10-20    Bernard       The first version
9  */
10
11 #include <stdio.h>
12
13 int main(int argc, char **argv)
14 {
15     printf("hello world!\n");
16
17     return 0;
18 }
```

```
bernard@bernard-MC8178EH:~/workspace/alliance/rtthread-smart/kernel/bsp/qemu-vexpress-a9$ ./qemu-nographic.sh
WARNING: Image format was not specified for 'sd.bin' and probing guessed raw.
Automatically detecting the format is dangerous for raw images, write operations on block 0 will be restricted.
Specify the 'raw' format explicitly to remove the restrictions.
pulseaudio: set_sink_input_volume() failed
pulseaudio: Reason: Invalid argument
pulseaudio: set_sink_input_mute() failed
pulseaudio: Reason: Invalid argument

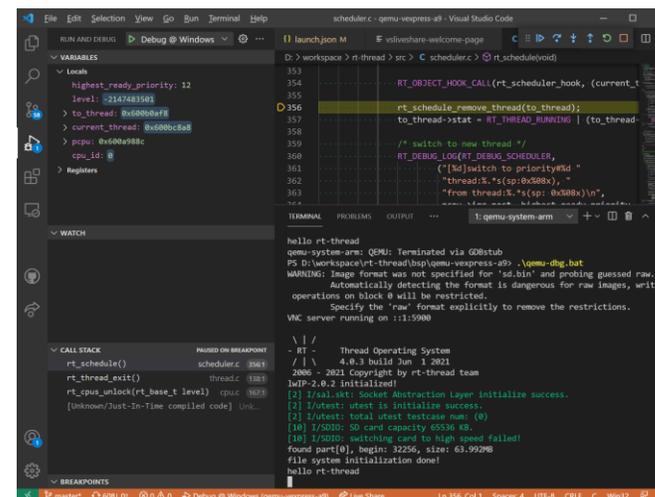
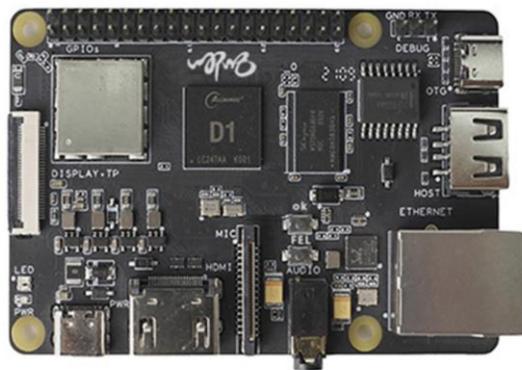
// /
- RT -   Thread Smart Operating System
/ | \   - 5.0.0 build Jun  5 2023
2006 - 2020 Copyright by rt-thread team
lwIP-2.0.2 initialized!
try to allocate fb... | w - 640, h - 480 | done!
fb -> 0xd1100000
[I/so1.9k] Socket Abstraction Layer initialize success.
[I/S010] SD card capacity 65536 KB.
File system initialization done!
hello rt-thread
msh /mnt/bin/hello
msh ./hello world!
```



# RT-Thread Smart 调试手段

支持 **gdb** 调试用户态应用程序:

- 不需要硬件仿真器
- 通过串口连接调试
- 通过网口连接调试
- 支持条件断点, 支持attach到应用程序调试



UART/Networking



# RT-Thread Smart vs RT-Thread

应用:

➡ 基本无感, 有几点需要注意的:

`rt_thread_create`, 栈大小指的是用户态栈大小; 内核态栈由系统指定

`rt_mb_init/rt_mq_init`, 提供的 (用户态) pool空间将无效, 而由系统在内核中指定

驱动:

➡ 由于系统都运行在虚拟地址上, 对外设的访问也不再是实地址, 而是虚地址模式:

需要通过 `void *rt_ioremap(void *paddr, size_t size)` 映射成虚拟地址使用



# RT-Thread Smart 微内核/混合内核 架构

## rt-smart 是一种混合架构的操作系统

- 内核运行在虚拟地址，外设地址需要以映射后的虚拟地址访问
- 系统服务，例如文件系统，网络协议栈放回到内核中时，它是完全宏内核方式运行
- 当把应用都放在内核中执行时，类似传统的RT-Thread <除了虚地址的差别，但在Cortex-A上本身也会配置成1:1映射关系>

## 本身具备channel + 共享内存的消息机制

## 微内核能力体系在：

- 一些系统服务可以外置在用户态；内核中通过简单代理方式，消息传递到核外系统服务
- 依然具备非常全的系统调用，即用户态应用会先以系统调用方式和内核交互，然后内核再决定具体服务方
- 提供channel + 共享内存方式的消息机制，也可以用于直接方式的用户态应用  系统服务模式



  
RT-Thread

THANK



YOU

网站: [www.rt-thread.org](http://www.rt-thread.org) 微信: RTThread