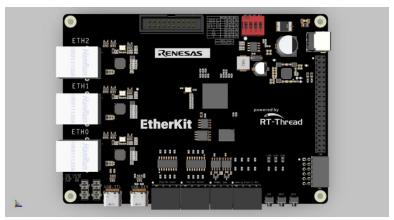


RZN2L Quick Start----基于 Etherkit 开发板

简介

本应用笔记介绍了如何使用 RZ/N2 Etherkit 开发板快速入门。描述硬件配置和 IDE IAR 和 E2studio 软件环境搭建。



开发工具

• IDE: IAR EW for Arm 9.50.2 或 E2studio 2024-01.1

FSP: RZ/N2 FSP V2.0

• 仿真器: Jlink V12

实验材料

- Etherkit 开发板
- Jlink 仿真器,需支持瑞萨 R52 内核

实验部分

1	硬件设置	2
2	IAR 环境开发入门	3
2	F2ctudio 环培开发入门	-



1 硬件设置

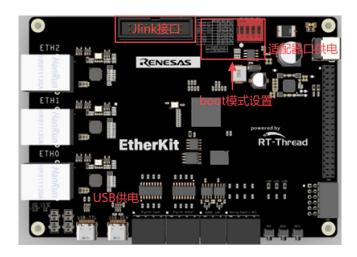
本节介绍使用 IAR 环境 和 E2studio 软件安装和环境配置。

1.1 开发板设置:

● 供电:可选 USB 供电或适配器供电

● Boot 模式设置:推荐 xSPI0 x1 boot mode

Jlink v12



本节完



2 IAR 环境开发入门

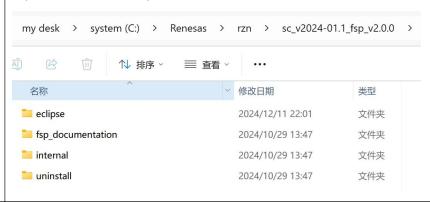
本节介绍使用 IAR 环境软件安装和环境配置。

2.1 IAR 软件安装:

- 安装 IAR EW for Arm 9.50.2 IAR 安装不做详细说明
- 安装 FSP 2.0: RZN2L: setup rznfsp v2 0 0 rzsc v2024-01.1.exe

打开 setup_rznfsp_v2_0_0_rzsc_v2024-01.1.exe 安装软件,安装后的目录如下图:

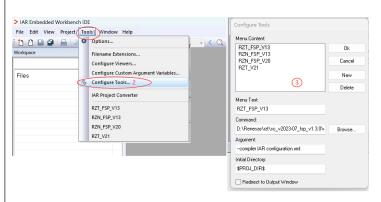
FSP 软件使用说明参考 fsp_documentation 文件夹下面的说明文件,里面每个外设参数都有说明,还有一些例程参考。



2.2 FSP 导入 IAR

FSP 可以作为 Tools 导入到 IAR 环境,方便后续随时打开 FSP,进行配置修改更新。

打开 IAR-Tools-Configure Tools...如下图:

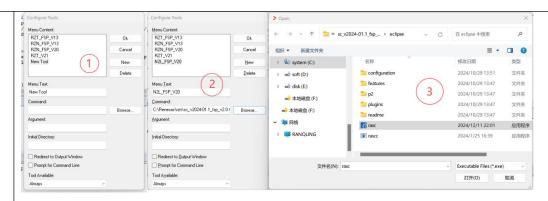


从上图可以看到,已经导入多个版本 FSP,新增加点击 New,

出现如下图界面 1,出现 New Tools,修改名称为: N2L FSP V20。

然后点击 Browse,选择 FSP 安装目录,选定 rase 文件,点击打开。

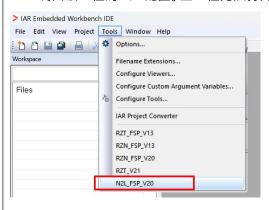




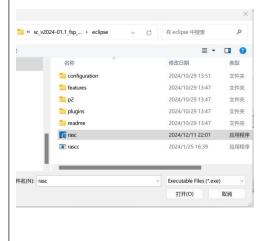
Argument 输入: --compiler IAR configuration.xml

Initial Directory 输入: \$PROJ_DIR\$

- 点击 OK, 即添加成功。
- IAR 点击 Tools 即可看到刚添加的 FSP 工具 N2L_FSP_V20,打开一个例程,点击该工具,即可调出该工程的 FSP 配置。空工程无法打开该工具。

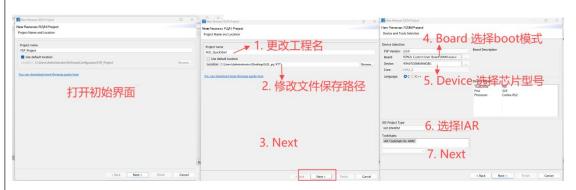


2.3 ● 利用 FSP 新建 IAR 工程





● 打开 rasc.exe-- File > New > FSP Project...更改工程名、文件路径--下一步--boot 模式、芯片型号、IDE 选择--下一步



- Boot 模式选择:RZN2L Custom User Board(xSPI0 x1 boot mode)
- Next 进入到如下界面:



2.4 FSP 功能窗口简介

- BSP 窗口: 芯片型号 boot 模式选择 芯片信息
- Clocks 窗口: 时钟树设置
- Pins 窗口: pin 脚功能设置

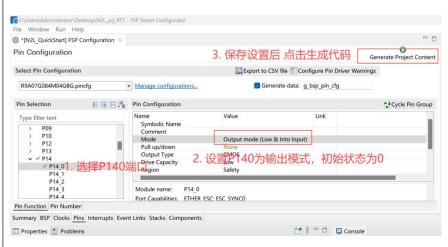




2.5 | 生成 IAR 代码

● 生成一个简单的代码,设置 P140 为输出,控制 LED 灯。

设置 P140 为输出模式,初始状态为低电平,然后点击 Generate Project Content 生成代码。



进入工程目录,打开IAR工程:

下图为 FSP 代码目录结构, void hal_entry(void) 为入口函数,用户代码在此添加。



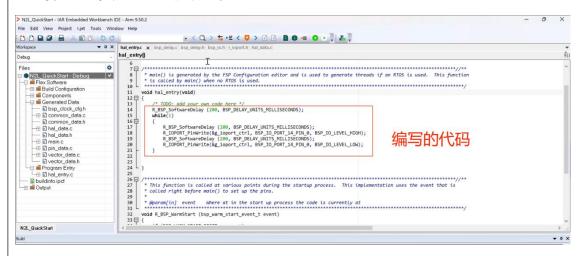




编写代码

● 利用系统生成延时函数:

void R_BSP_SoftwareDelay (uint32_t delay, bsp_delay_units_t units),控制 P140 电平输出,使 LED1 亮灭 200ms 反转一次。



2.8 Xspi0 boot 模式 debug 注意事项

在 xSPI0 boot 模式下 debug 的时候,程序不会停止用户入口函数处,需要在工程编译前插入如下代码到 startup core.c:

BSP_TARGET_ARM void system_init (void)

{

#if 1 // Software loops are only needed when debugging.

asm volatile (

" mov r0, #0 \n"

" movw r1, #0xf07f \n"

" movt r1, #0x2fa \n"

"software_loop: \n"

" adds r0, #1 \n"

" cmp r0, r1 \n"

"bne software_loop \n"

::: "memory");

#endif

如果第一次进入仿真界面,程序没有停止在用户程序入口 int main()位置,需暂停程序,IAR 快捷 RESET 设置为: Hardware(Reset pin),点击 RESET 后,程序就会停在函数入口 int main()位置。如下图:



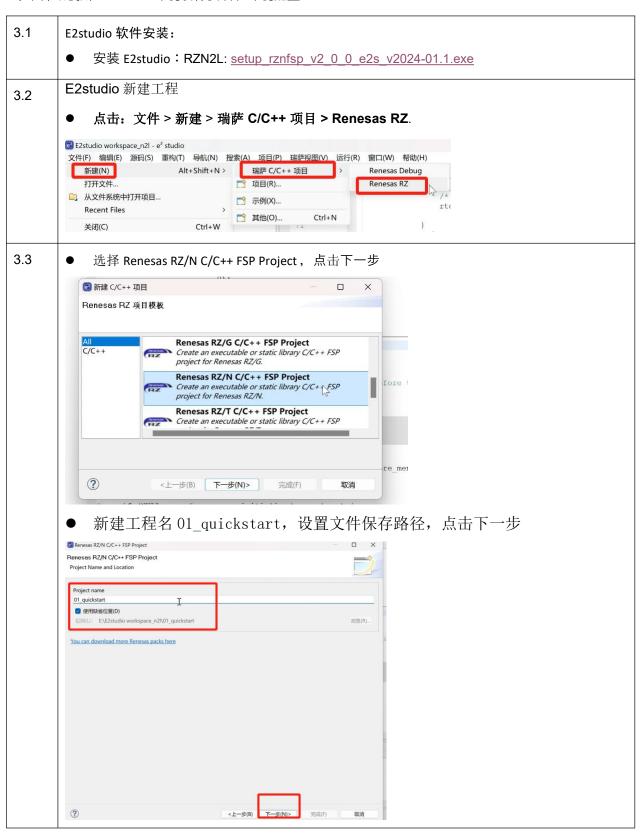


本节完

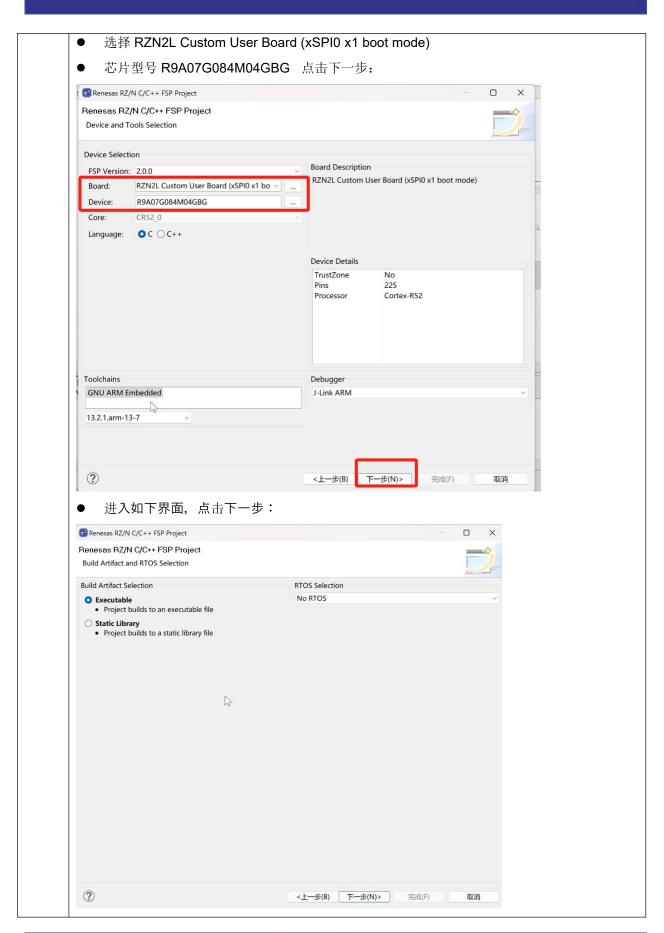


3 E2studio 环境开发入门

本节介绍使用 E2studio 环境软件安装和环境配置。





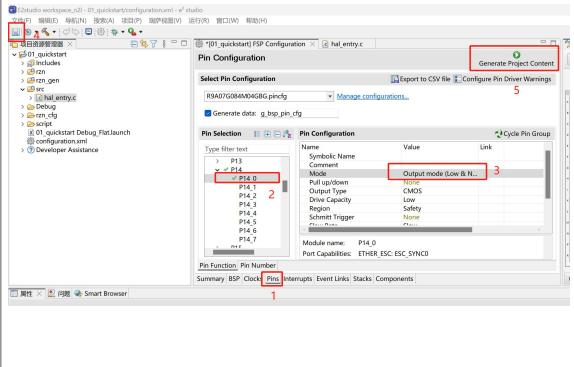














编写用户代码 利用系统生成延时函数: void R_BSP_SoftwareDelay (uint32_t delay, bsp_delay_units_t units), 控制 P140 电平输 出, 使 LED1 亮灭 200ms 反转一次。 打开 hal entry.c 进入 void hal entry(void) 为入口函数,添加用户代码。 R_BSP_SoftwareDelay (200, BSP_DELAY_UNITS_MILLISECONDS); while(1) { R BSP SoftwareDelay (200, BSP DELAY UNITS MILLISECONDS); R_IOPORT_PinWrite(&g_ioport_ctrl, BSP_IO_PORT_14_PIN_0, BSP IO LEVEL HIGH); R BSP SoftwareDelay (200, BSP DELAY UNITS MILLISECONDS); R IOPORT PinWrite(&g ioport ctrl, BSP IO PORT 14 PIN 0, BSP IO LEVEL LOW); } → 第 O1 quickstart [Debug]
→ 第 二进制
→ 前 Includes
→ 達 アzn FSP_CPP_HEADER

woid R_BSP_WarmStart(bsp_warm_start_event_t event)

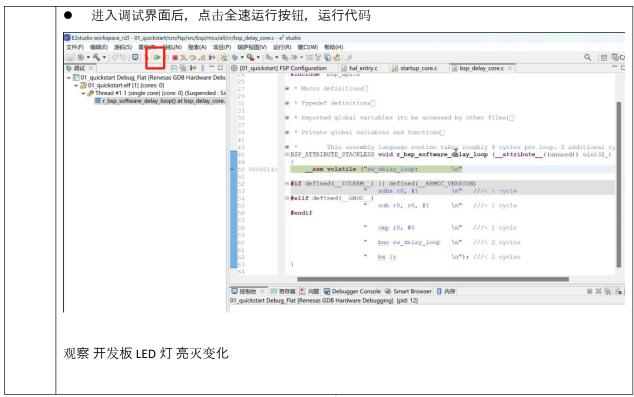
BSP_PLACE_IN_SECTION(".warm_start");

FSP_CPP_FOOTER rzn gen > c hal_entry.c
> Debug \oplus * main() is generated by the FSP Configuration editor and is used to generate threads if an \ominus void hal_entry(void) ⇒build 1 ⇔rzn_cfg ⇔scrir* 1 01_quickstart Debug_Flat.launch rzn_cfg.txt
> ② Developer Assistance ⊕ * This function is called at various points during the startup process.
⊕ void R_BSP_WarmStart(bsp_warm_start_event_t event) | 属性 🎅 问题 🌑 Smart Browser 📮 控制台 > CTO Build Console [01 quickstart.elf
Building target: 01_quickstart.elf
Building target: 01_quickstart.elf
arm-none-eabi-objcopy -0 srec "01_quickstart.elf" "01_quickstart.srec"
arm-none-eabi-size --format-berkeley "01_quickstart.elf"
text data bss dec hex filename
5624 404 20304 26332 66dc 01_quickstart.elf 编译无报错 11:24:54 Build Finished. 0 errors, 0 warnings. (took 2s.509ms) 3.6



调试:开发板 Jlink 正确连接, 上电。 新工程第一次调试,需进行调试配置,配置方法如下: E2studio workspace_n2l - 01_quickstart/src/hal_entry.c - e² studio 文件(F) 编辑(E) 源码(S) 重构(T) 导航(N) 搜索(A) 项目(P) 瑞萨视图(V) 运行(R) 窗口(W) 帮助(H) → 项目资源管理器 × (无启动历史记录) [01_quickstart] FSP Configuration #include "hal_data.h" 调试方式(D) > 🍇 二进制 调试 配置(B)... FSP_CPP_HEADER > 🔝 Includes > 🔑 rzn void R BSP WarmStart (bsp_warm_s BSP_PLACE_IN_SECTION(".warm_sta 组织收藏夹(V)... > <a> rzn_gen FSP_CPP_FOOTER ✓ 🕮 src > lc hal_entry.c ⊕ * main() is generated by the F > 🗁 Debug ⊖ void hal_entry(void) > 🗁 build > > rzn_cfg /* TODO: add your own code R_BSP_SoftwareDelay (200, E 🕅 01 quickstart Debug Flat.launch 打开调试配置 选择我们创建的工程, 然后点击 调试 按钮 🕝 调试 配置 创建、管理和运行配置 * F = X = F + 名称(N): 01_quickstart Debug_Flat 输入筛选文本 □ 主要 ‡ 调试器 ▶ Startup ‡ 源 □ 公共 © C/C++ Remote Application © C/C++ 应用程序 ☑ EASE Script © GDB Hardware Debugging 项目(P): 01 quickstart 浏览(B). C/C++ 程序: GDB Simulator Debugging (RH850)
Renesas GDB Hardware Debugging
G 01 quickstart Debug Flat Debug/01_quickstart.elf 搜索项目(H)... 浏览(R) Build (if required) before launching ■ 启动组 Build Configuration: Use Active O Enable auto build O Disable auto build ○ 使用工作空间设置 配置工作空间设置... 恢复(V) 过滤器已匹配 9 项, 总共 11 项 ? 调试(D) 关闭 跳出如下窗口,点击切换按钮 📴 确认切换透视图 X 这种启动被配置为在暂停时打开调试视图。 这个"调试"透视图用于支持应用程序调试。它包含用于显示调试堆栈、显示变量和管 理断点的视图。 切换到这个视图? □ 记住我的决定(R) 切换(S) 否(N)





本节完