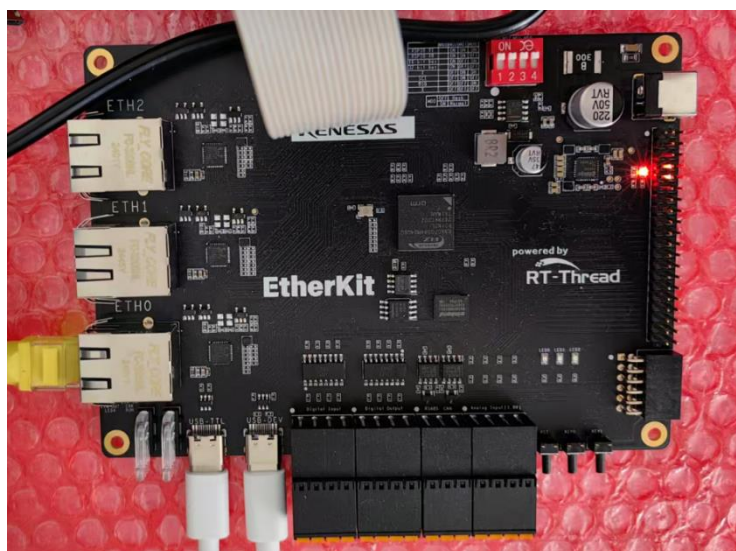


RZN2L IIC 例程操作手册-----基于 Etherkit 开发板

简介

本应用笔记介绍了基于 RZ/N2 Etherkit 开发板的 IIC 操作板载 EEPROM : AT24C16。分别介绍 IDE IAR 和 E2studio 软件下的操作。



开发工具 <ul style="list-style-type: none"> • IDE: IAR EW for Arm 9.50.2 E2studio 2024-01.1 • FSP: RZ/N2 FSP V2.0 • 仿真器: Jlink V12 	实验材料 <ul style="list-style-type: none"> • Etherkit 开发板 • Jlink 仿真器, 需支持瑞萨 R52 内核
--	---

实验部分

1.硬件设置及软件安装	2
2 .IAR 环境工程介绍.....	3
3 .E2studio 环境工程介绍	9

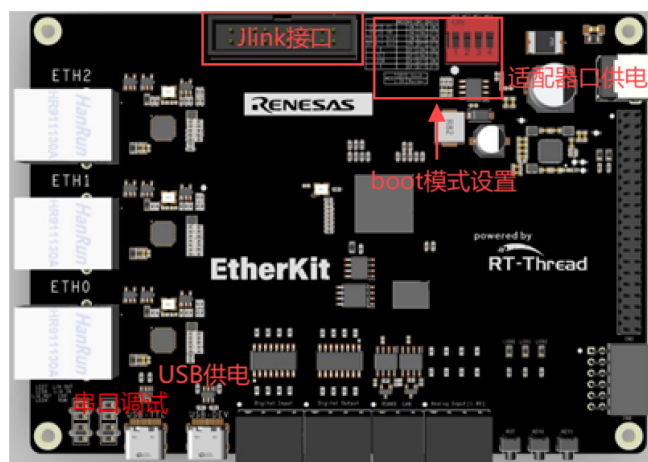
1 .硬件设置及软件安装

本节 EtherKit 开发板硬件设置。

1.1

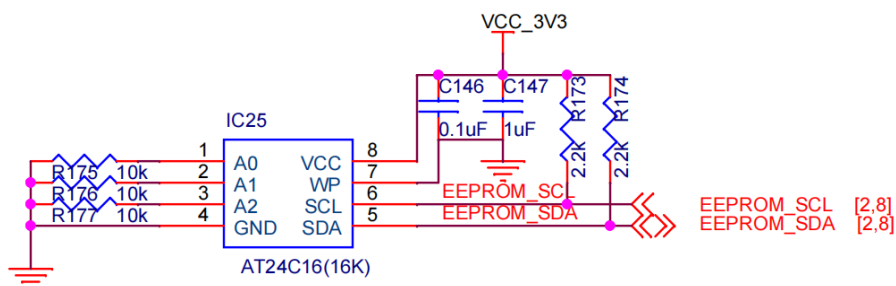
开发板设置：

- 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



1.2

硬件原理图：



EtherKit 上的 EEROM 使用为 AT24C16 连接 芯片的 IIC0；

本节完

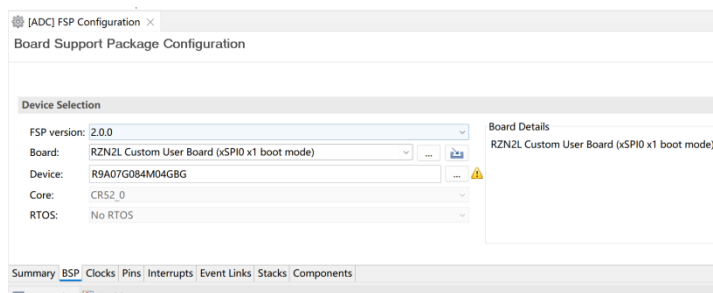
2 .IAR 环境工程介绍

本节介绍 IAR 环境下 IIC 工程。

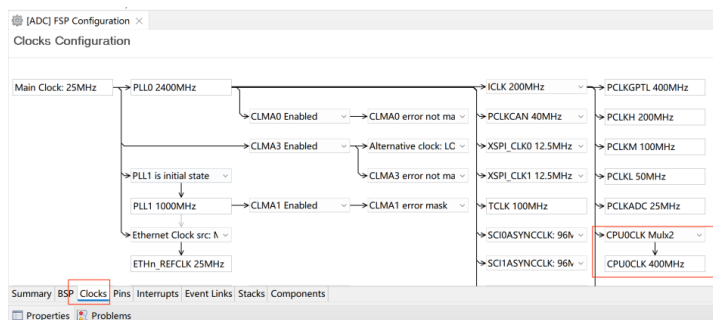
2.1

● 打开 FSP 新建工程: IIC

1. 选择 RZN2L Customer User Board (xSPI0 X1 boot mode)

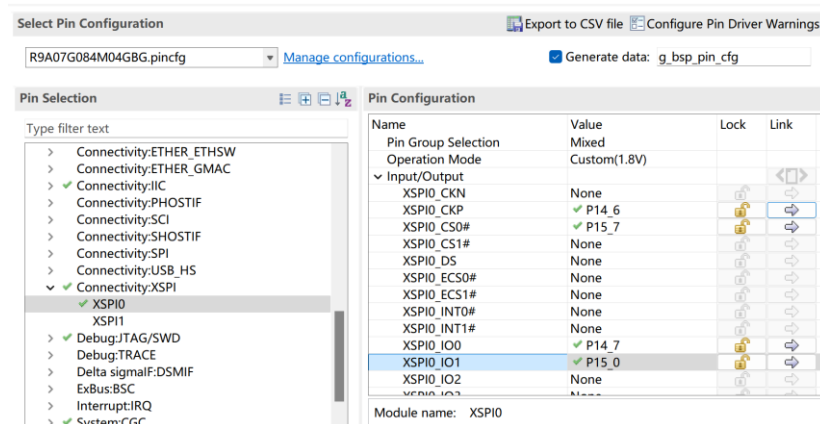


2. 主时钟设置 400MHZ

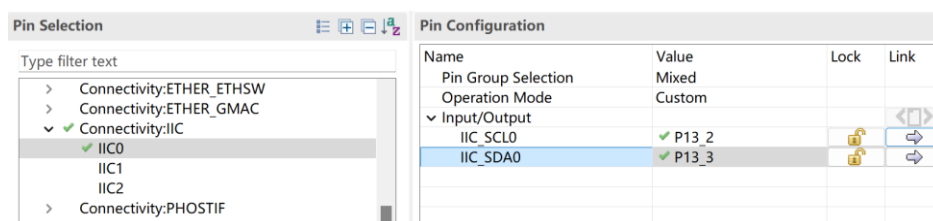


3. 配置 XSPI0

Pin Configuration

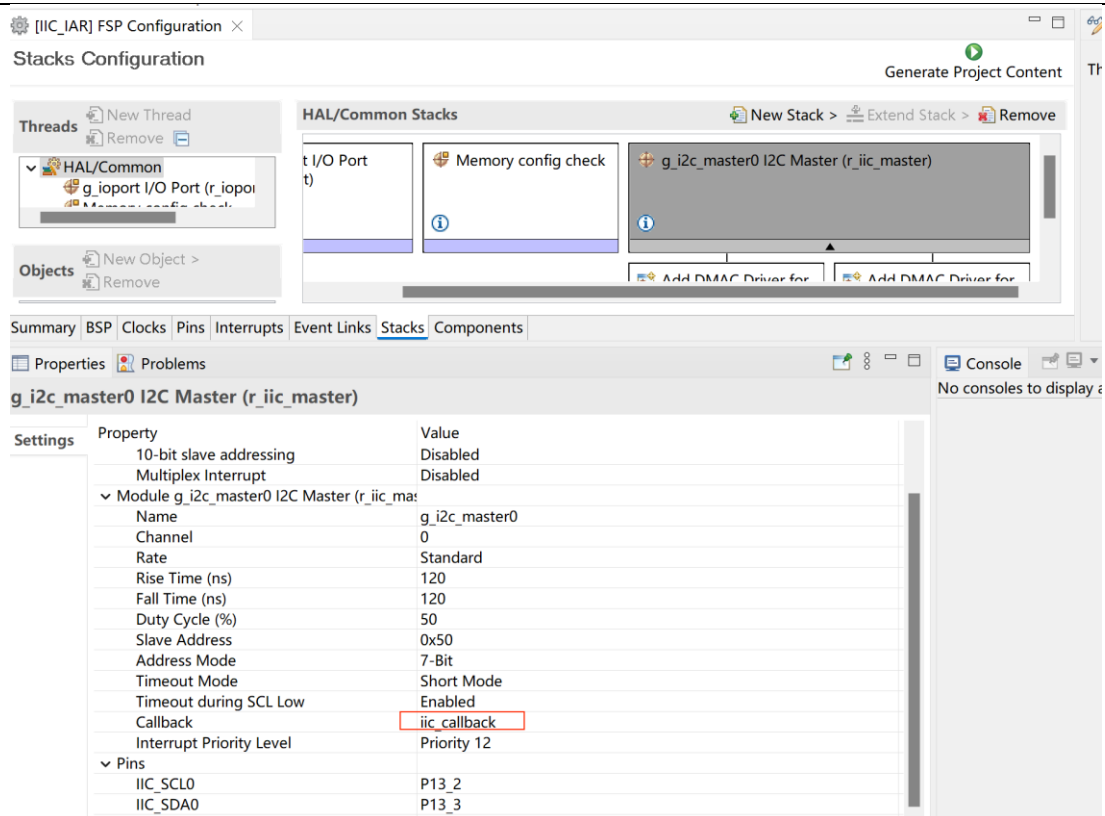


4. IIC 引脚配置

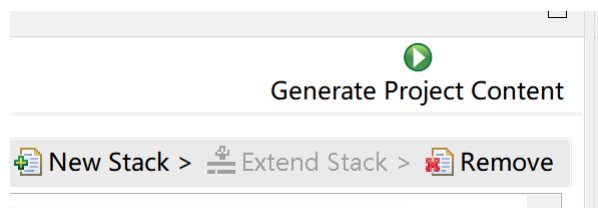


5. 添加 IIC master Stack

添加回调函数 iic_callback, 从机地址 设置 0X50 = EEPROM 7 位地址

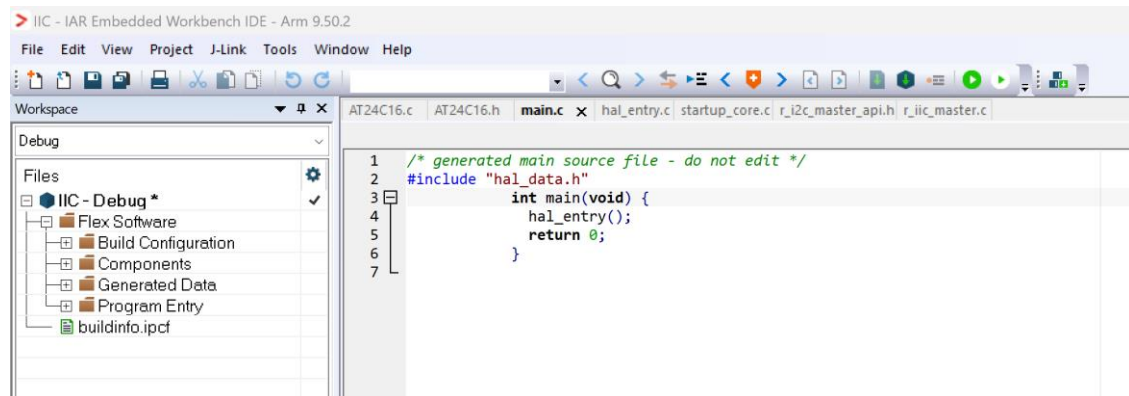


6. 点击: Generate Project Content 生成代码

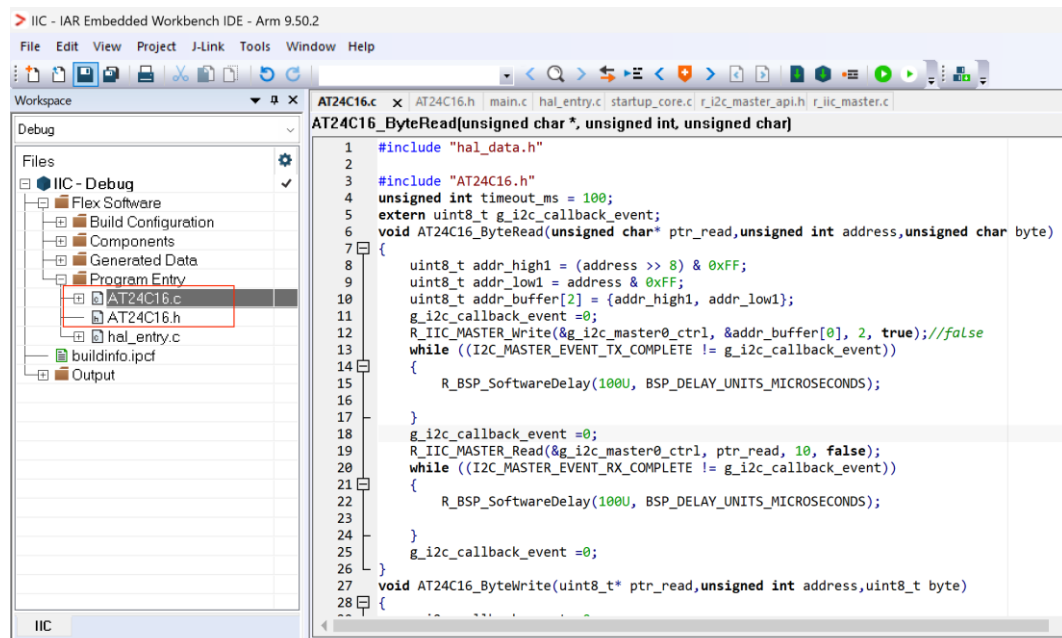


2.2

7. 打开生成的代码



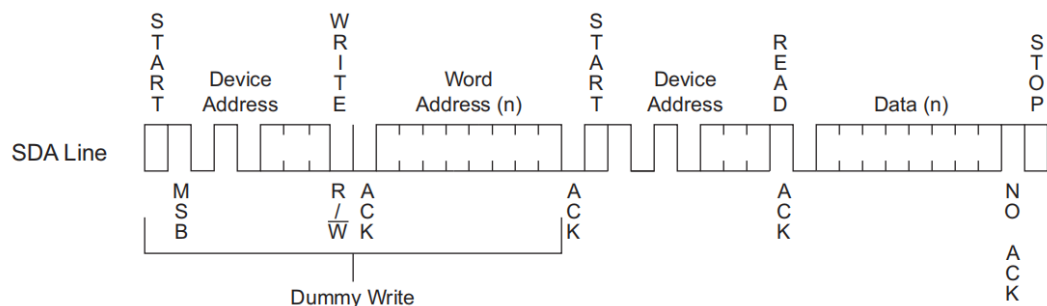
- 仿真器由 Ijet 切换为 Jlink
- 编写用户代码



➤ void AT24C16_ByteRead(unsigned char* ptr_read, unsigned int address, unsigned char byte) ;

实现随机地址读数据，实现 EEPROM 以下读时序：

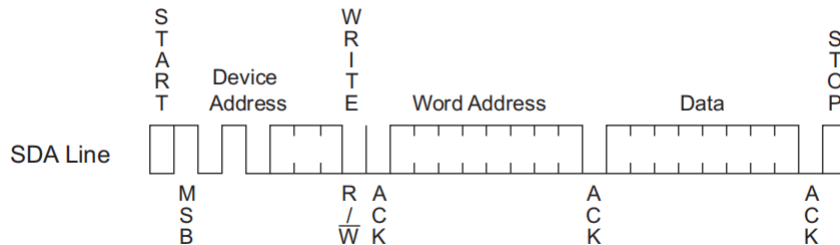
Figure 9-2. Random Read



➤ void AT24C16_ByteWrite(uint8_t* ptr_read,unsigned int address,uint8_t byte) ;

实现随机地址写数据，实现 EEPROM 以下写时序：

Figure 8-1. Byte Write



- 回调函数编写，IIC 事件赋值给 g_i2c_callback_event

```
void iic_callback (i2c_master_callback_args_t * p_args)
```

```
{
    g_i2c_callback_event = p_args->event;
}
```

- 测试代码编写：写数据到 EEROM，然后读回来，看是否正确。

```
AT24C16.c AT24C16.h main.c hal_entry.c x startup_core.c r_i2c_master_api.h r_iic_master.c
hal_entry()
22 fsp_err_t err;
23 uint8_t i;
24 //uint16_t write_addr;
25 //uint8_t read_data = 0x00;
26 uint8_t read_buffer[20]={};
27
28 err = R_IIC_MASTER_Open(&g_i2c_master0_ctrl1, &g_i2c_master0_cfg);
29 if (FSP_SUCCESS != err)
30 {
31     while(1);
32 }
33 R_BSP_SoftwareDelay (10, BSP_DELAY_UNITS_MILLISECONDS);
34 __asm volatile ("cpsie i");
35 __asm volatile ("isb");
36
37 for(i=2;i<12;i++)
38 {
39     tx_buffer[i]=i-1;
40 }
41 AT24C16_ByteWrite(&tx_buffer[0],0x0000,12);
42
43 for(i=0;i<10;i++)
44 {
45     read_buffer[i]=0;
46 }
47
48 R_BSP_SoftwareDelay (100, BSP_DELAY_UNITS_MILLISECONDS);
49 AT24C16_ByteRead(&read_buffer[0],0x0000,10);
50
51 while(1)
52 {
53
54
```

- Rebuild All---编译工程 无报错

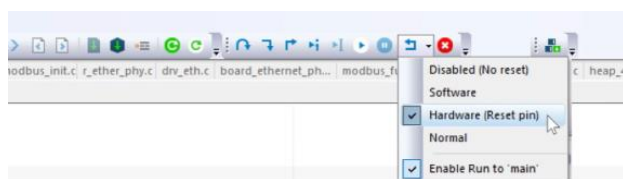
uild

Messages

```
Total number of errors: 0
Total number of warnings: 0
Resolving dependencies...
Build succeeded
```

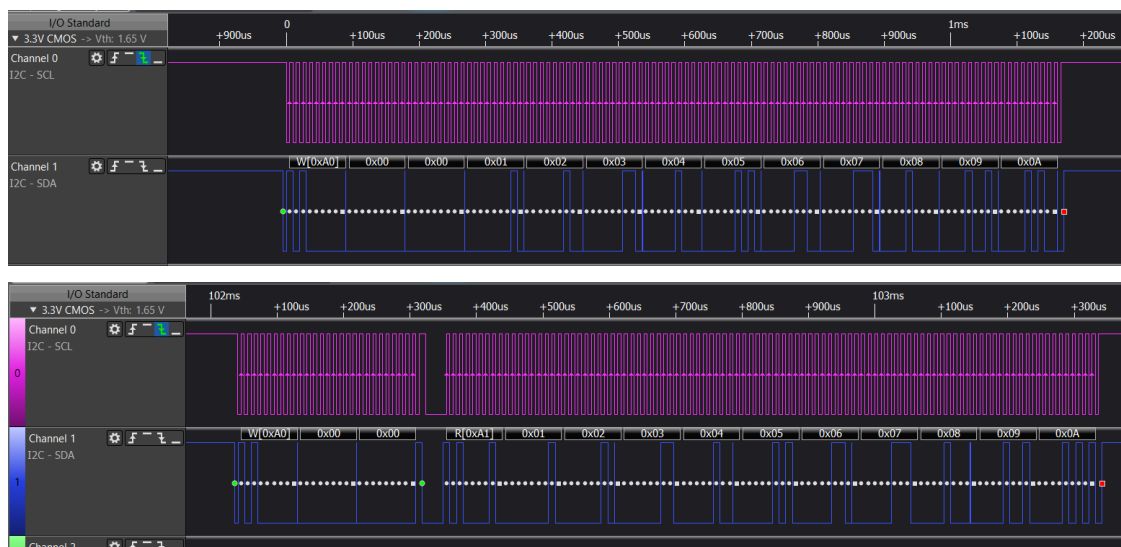
2.3

- Download and Debug --- 下载程序
- 下载工程到开发板，进入仿真界面
- Debug 复位设置为 Hardware



- 全速运行代码，以下是逻辑分析仪抓到的写数据和读数据时序。

读到的数据和写入一致。



本节完

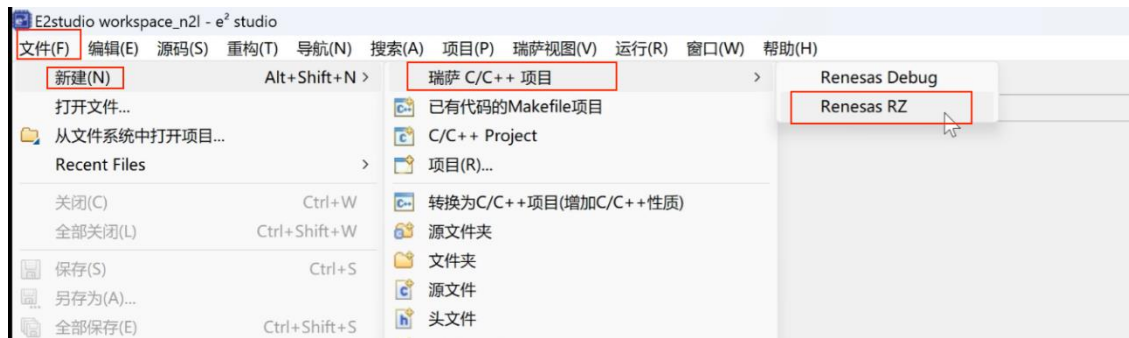
3 .E2studio 环境工程介绍

本节介绍使用 E2studio 环境创建 IIC 工程。

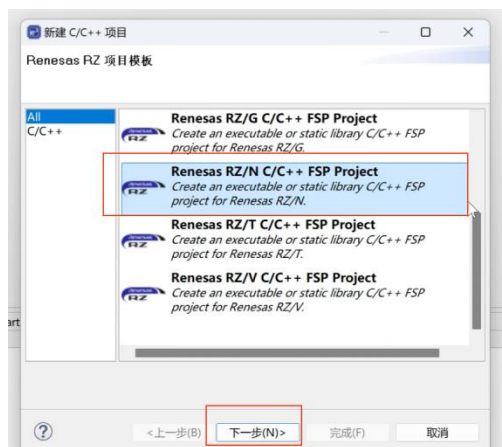
3.1

● 打开 E2studio, 新建工程

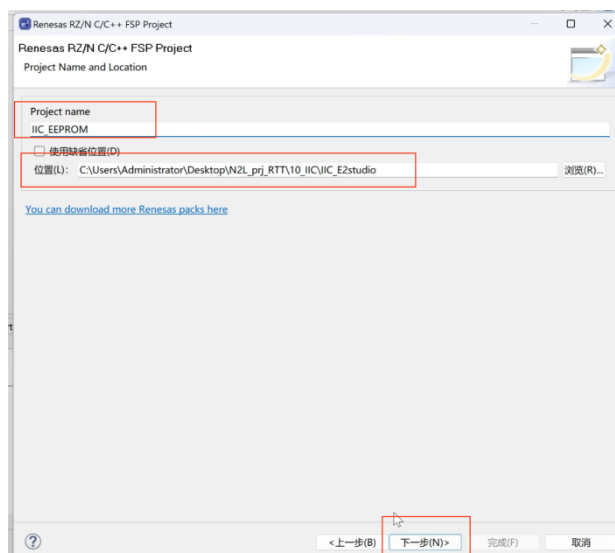
1. 选择 文件--新建--瑞萨 C/C++项目--Renesas RZ:



2. 选择 Renesas RZ/N C/C++ FSP Project

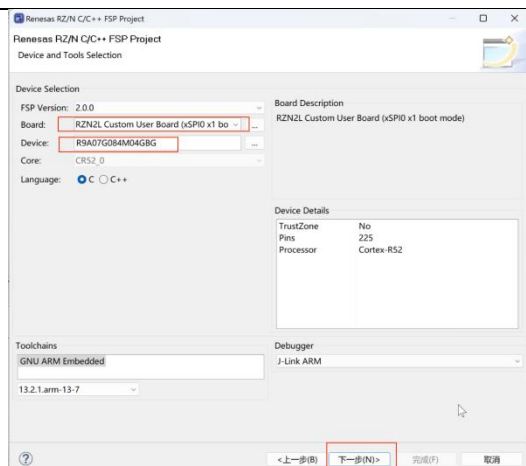


3. 设置项目名称: IIC_EEPROM, 选择工程保存路径

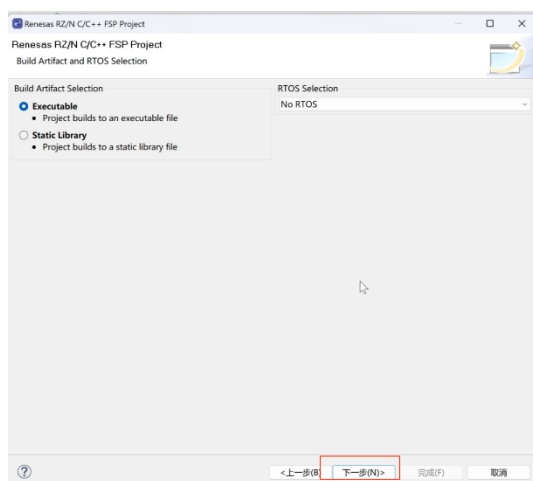


4. Boot 模式选择 RZN2L Custom User Board (XSPI0 x1 boot mode)

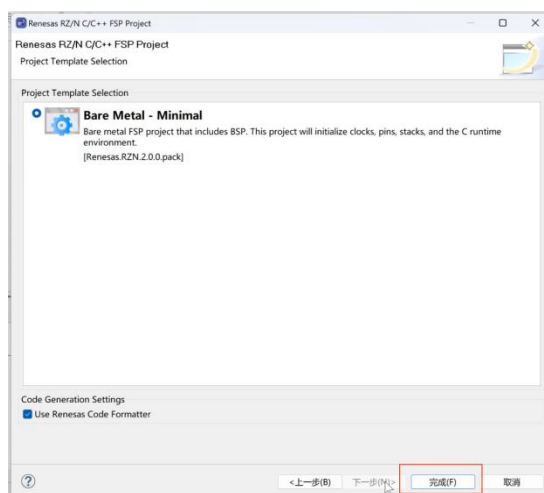
芯片型号: R9A07G084M04GBG



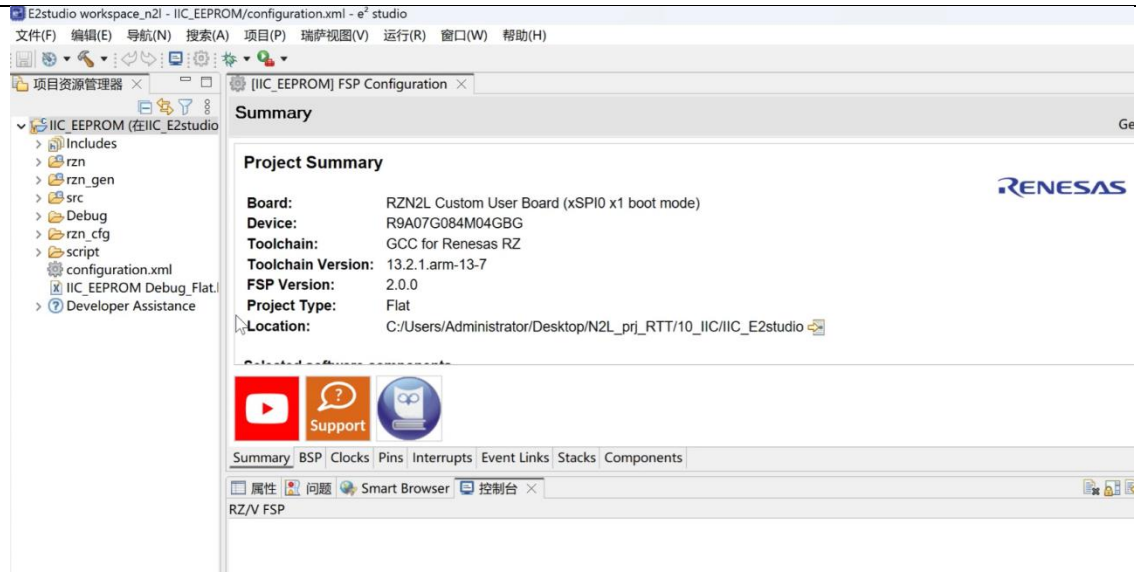
5. 直接下一步



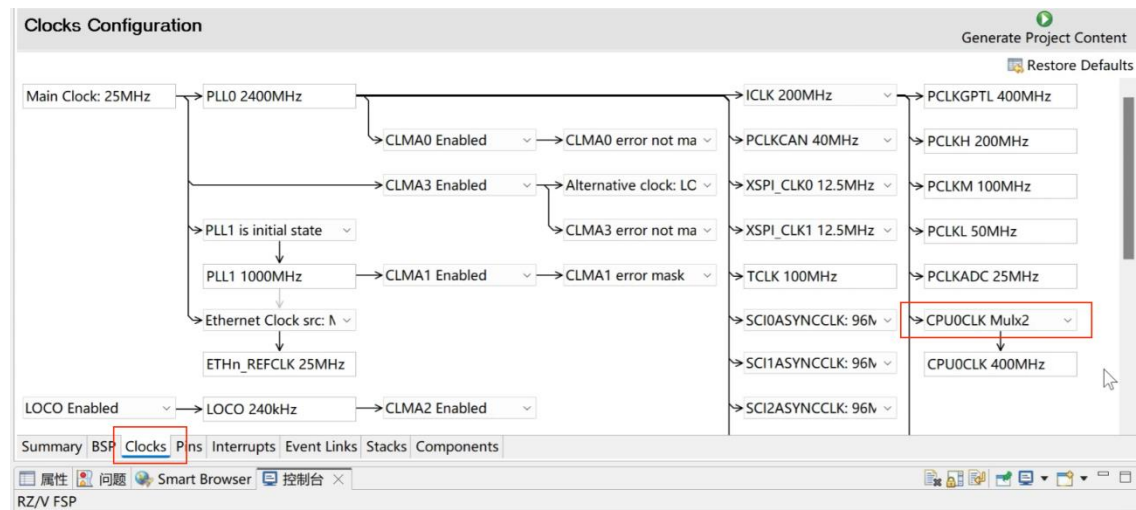
6. 点击 完成



7. 进入到工程界面

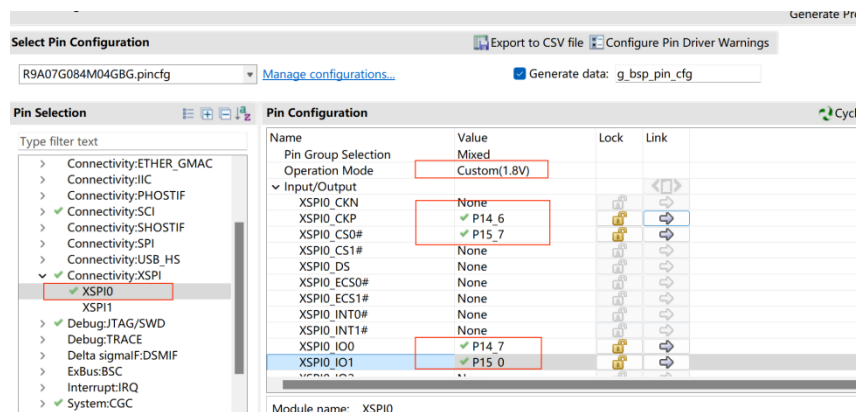


8. 设置时钟 CPU0CLK 400MHZ



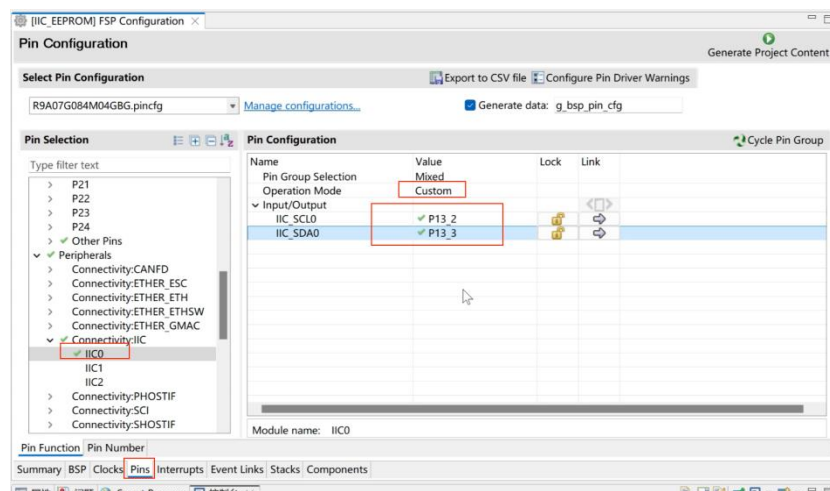
3.2

9. 配置 XSPI0

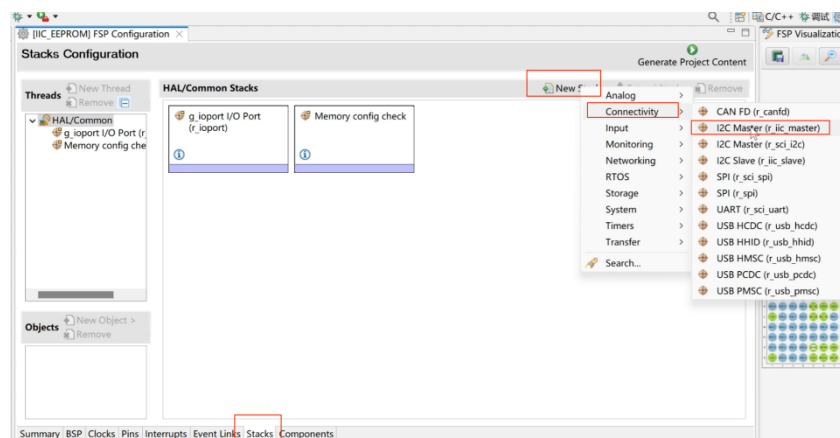


10. IIC 引脚配置

由原理图可知 EEPROM 接到 IIC0，打开 IIC0 外设，配置引脚。



8. 添加 NEW Stack: IIC master Stack



9. 设置回调函数 iic_callback，从机地址 0X50

HAL/Common Stacks

- g_ioport I/O Port (r_ioport)
- Memory config check
- g_i2c_master0 I2C Master (r_iic_master)

Summary | **BSP** | **Clocks** | **Pins** | **Interrupts** | **Event Links** | **Stacks** | **Components**

g_i2c_master0 I2C Master (r_iic_master)

Settings	属性	值
API Info	10-bit slave addressing	Disabled
	Multiplex Interrupt	Disabled
Module g_i2c_master0 I2C Master (r_iic_master)		
	Name	g_i2c_master0
	Channel	0
	Rate	Standard
	Rise Time (ns)	120
	Fall Time (ns)	120
	Duty Cycle (%)	50
	Slave Address	0x50
	Address Mode	7-Bit
	Timeout Mode	Short Mode
	Timeout during SCL Low	Enabled
	Callback	iic_callback
	Interrupt Priority Level	Priority 12
Pins		
	IIC_SCL0	P13_2
	IIC_SDA0	P13_3

10. 设置完毕，点击 **Generate Project Content** 生成代码

11. 用户代码编写

添加用户代码到工程，

```

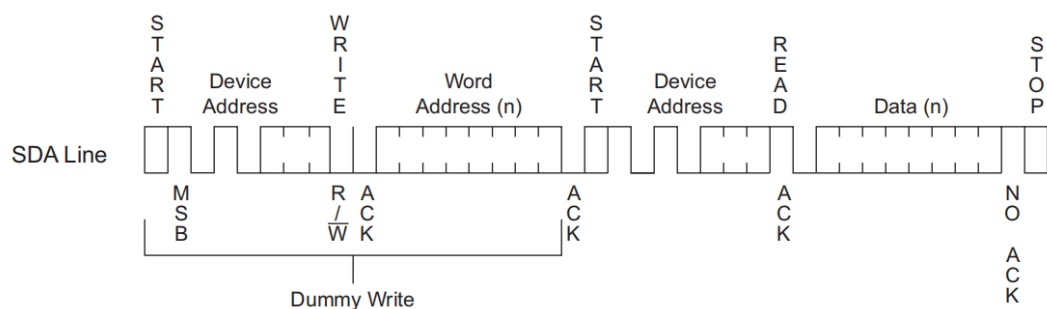
19 void hal_entry(void)
20 {
21     /* TODO: add your own code here */
22     fsp_err_t err;
23     uint8_t i;
24     uint8_t read_buffer[20]={};
25
26     err = R_IIC_MASTER_Open(&g_i2c_master0_ctrl, &g_i2c_master0_cfg);
27     if (FSP_SUCCESS != err)
28     {
29         while(1);
30     }
31     R_BSP_SoftwareDelay(100, BSP_DELAY_UNITS_MILLISECONDS);
32     __asm volatile ("cpsie i");
33     __asm volatile ("isb");
34     for(i=2;i<12;i++)
35     {
36         tx_buffer[i]=i-1;
37     }
38     AT24C16_ByteWrite(&tx_buffer[0],0x0000,12);
39     for(i=0;i<10;i++)
40     {
41         read_buffer[i]=0;
42     }
43     R_BSP_SoftwareDelay(100, BSP_DELAY_UNITS_MILLISECONDS);
44     AT24C16_ByteRead(&read_buffer[0],0x0000,10);
45
46     while(1)
47     {

```

➤ void AT24C16_ByteRead(unsigned char* ptr_read,unsigned int address,unsigned char byte) ;

实现随机地址读数据，实现 EEPROM 以下读时序：

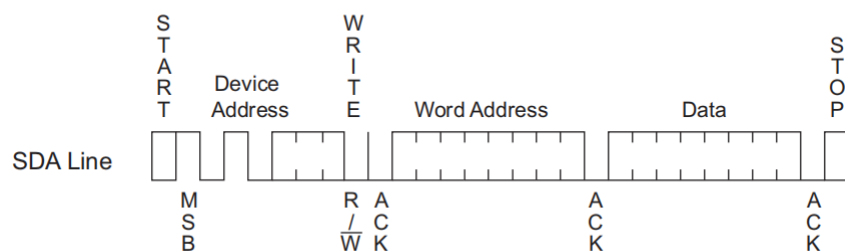
Figure 9-2. Random Read



➤ `void AT24C16_ByteWrite(uint8_t* ptr_read,unsigned int address,uint8_t byte) ;`

实现随机地址写数据，实现 EEPROM 以下写时序：

Figure 8-1. Byte Write



3.3

- 下载代码到开发板

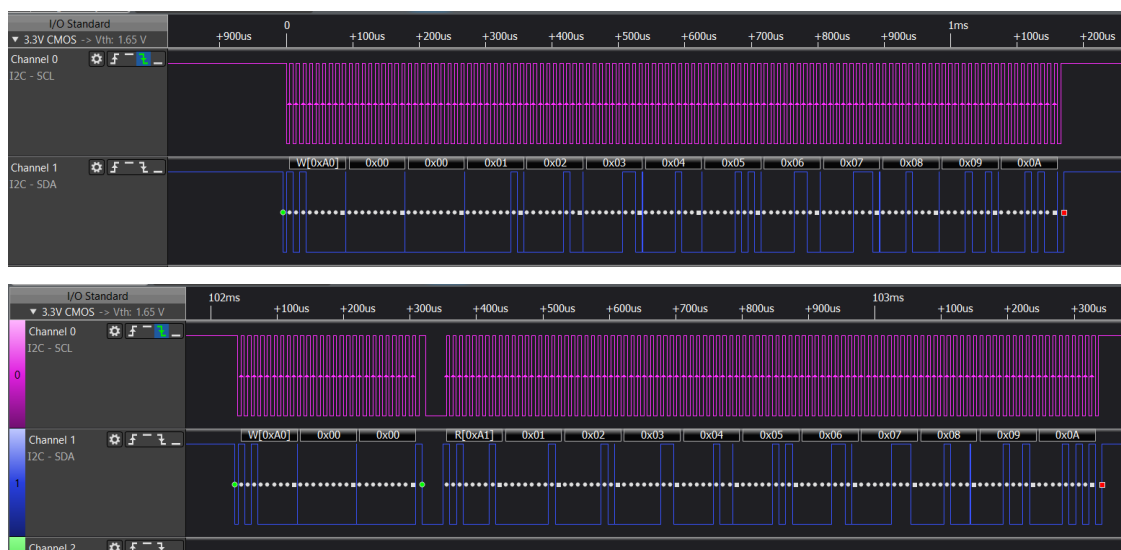
编译代码，无报错。

E2studio 在线仿真下载，全速运行

3.4

- 全速运行代码，以下是逻辑分析仪抓到的写数据和读数据时序。

读到的数据和写入一致。



本节完