栏目	描述
软件 IDE	1. KEIL MDK523 2. IAR EWARM-CD-8324-20889 3. GCC arm-gnu-toolchain-12.2.mpacbti-rel1-mingw-w64-i686-arm-none-eabi 4. Make xpack-windows-build-tools-4.4.0-1-win32-x64 5. CMake cmake-3.27.3-windows-x86_64
调试器	JLink_Windows_V614b

SPINTROL

目录

版本历史 5

1 MCU Power 6

1.1 MCU 电源过压/欠压触发中断 6

1.2 MCU 电源过压/欠压触发复位 6

2 Clock 7

2.1 时钟错误触发中断 7

2.2 时钟错误触发复位 7

2.3 PLL 时钟源选择 XO 或者 RCO 7

3 GPIO 8

3.1 GPIO 电平翻转检测 8

3.2 GPIO 电平高低检测 8

4 CRC 9

4.1 CRC 正常工作检测 9

5 timer 9

5.1 Timer 事件计数器 9

5.2 Timer 门控计数器 9

5.3 Timer 触发中断 10

6 WDT 10

6.1 WDT 喂狗 10

6.2 WDT 触发中断 11

7 PWM 11

7.1 PWM 输出互补波形 11

7.2 PWM 输出单通道波形 11

7.3 PWM 输出三相波形 12

7.4 级联同步方式设置 PWM 相位差-首节点软件触发 12

7.5 非级联同步方式设置 PWM 相位差-软件触发 13

7.6 非级联同步方式设置 PWM 相位差-GPIO 触发 14

7.7 非级联同步方式设置 PWM 相位差-Timer 触发 14

7.8 PWM 触发 ADC 采集 15

7.9 PWM 信号封锁-GPIO 触发 15

7.10 PWM 信号封锁-过压触发 16

7.11 PWM 信号封锁-COMP 触发两路 TZ 17

SPINTROL

©2026 旋智电子科技（上海）有限公司

8 ECAP 17

8.1 ECAP APWM 模式 17

8.2 ECAP 单通道持续捕获-Absolute 模式 18

8.3 ECAP 单通道持续捕获-Delta 模式 18

8.4 ECAP 单通道一次性捕获-Delta 模式 18

8.5 ECAP 单通道一次性捕获- Absolute 模式 19

8.6 ECAP 双通道持续捕获-Delta 模式 19

9 ADC 20

9.1 ADC 单端采样-软件触发 20

9.2 ADC 单端采样-定时器触发 20

9.3 ADC 双端采样-软件触发 21

9.4 ADC 增益及偏移校准 21

9.5 ADC 开路检测 21

9.6 ADC 短路检测 22

10 TSENSER 23

10.1 Tsenser 测量 MCU 温度 23

11 PGA 23

11.1 PGA 校准 23

12 Comparator 24

12.1 设置比较器阈值及检测 24

13 DAC 25

13.1 DAC 正常工作检测 25

14 CRC 25

14.1 CRC 正常工作检测 25

15 UART 25

15.1 UART 自动波特率模式 25

15.2 UART 阻塞方式-发送接收检查 26

15.3 UART 阻塞方式-接收发送 26

15.4 UART 非阻塞中断方式-接收不定长数据 26

16 I2C 27

16.1 I2C 主机模式-阻塞方式 27

16.2 I2C 从机模式-阻塞方式 27

16.3 I2C 主机模式-非阻塞中断方式 28

16.4 I2C 从机模式-非阻塞中断方式 28

SPINTROL

17 SPI 29
17.1 SPI 主机模式-阻塞方式 29
17.2 SPI 从机模式-阻塞方式 29
17.3 SPI 主机模式-非阻塞中断方式 29
17.4 SPI 从机模式-非阻塞中断方式 30
18 FLASH 32
18.1 FLASH 模拟 EEPROM 32
18.2 FLASH 读写擦操作 32
18.3 扇区写/擦保护 32
18.4 在 RAM 中运行代码及数据 33

SPINTROL

©2026 旋智电子科技（上海）有限公司

版本历史

版本	日期	作者	状态	变更
C/0			Released	1.

SPINTROL

1 MCU Power

1.1 MCU 电源过压/欠压触发中断

栏目	描述
工程名	BOD_UVOV_Int
功能	MCU 电源过压/欠压触发中断
硬件环境	1. 需可调电源供电。
步骤	1. 设置 VDD3.3V 电源域的欠压检测，设置为 3.03V 时，欠压事件触发；3.12V 时，欠压事件消失。 设置 VDD3.3V 电源域的过压检测，设置为 3.79V 时，过压事件触发；3.66V 时，过压事件消失。 2. 使能 VDD3.3V 电源域触发的中断为不可屏蔽中断，填写中断服务函数，在函数中查询过压/欠压事件发生标志，并清除中断。
预期结果	1. 微调可调电源，当低于 3.03V 时，打印 VDD33 触发欠压中断，直至调整电源高于 3.12V 时，打印消失。 2. 微调可调电源，当高于 3.79V 时，打印 VDD33 触发过压中断，直至调整电源低于 3.66V 时，打印消失
注意点	

1.2 MCU 电源过压/欠压触发复位

栏目	描述
工程名	BOD_UVOV_Reset
功能	MCU 电源过压/欠压触发复位
硬件环境	1. 需可调电源供电。
步骤	1. 查询过压/欠压复位事件发生标志。 2. 设置 VDD3.3V 电源域的欠压检测，设置为 3.03V 时，欠压事件触发； 设置 VDD3.3V 电源域的过压检测，设置为 3.79V 时，过压事件触发； 3. 使能 VDD3.3V 电源域的过压/欠压事件触发复位。
预期结果	1. 微调可调电源，当低于 3.03V 时，打印 VDD33 触发欠压复位。 2. 微调可调电源，当高于 3.79V 时，打印 VDD33 触发过压复位。
注意点	1. 撤销过压/欠压复位事件的电压值为寄存器的默认值（撤销过压复位值为 4.08V，撤销欠压复位值为 2.65V）。

©2026 旋智电子科技（上海）有限公司

2 Clock

2.1 时钟错误触发中断

栏目	描述
工程名	Clock_Detect_Error_Int
功能	时钟错误触发中断
硬件环境	
步骤	1. 初始化时钟探测，配置探测的 PLL 时钟为 100MHz，误差为 1%。 2. 使能 PLL 失锁产生不可屏蔽中断，并填写中断服务函数，在函数中查询 PLL 失锁发生标志，并清除中断。 3. 使能时钟探测。
预期结果	1. 程序运行后，触发时钟探测错误中断，打印中断次数持续增加。
注意点	

2.2 时钟错误触发复位

栏目	描述
工程名	Clock_Detect_Error_Reset
功能	时钟错误触发复位
硬件环境	
步骤	1. 查询 PLL 失锁发生标志。 2. 使能 PLL 失锁产生复位。 3. 初始化时钟探测，配置探测的 PLL 时钟为 100MHz，误差为 1%。 4. 使能时钟探测。
预期结果	1. 程序运行后，先打印没有发生时钟错误事件。 2. 程序继续运行，产生失锁事件后，芯片复位。 3. 芯片复位后，打印发生时钟错误事件。
注意点	

2.3 PLL 时钟源选择 XO 或者 RCO

栏目	描述
工程名	Clock_Use_XO_Or_RCO
功能	PLL 时钟源选择 XO 或者 RCO
硬件环境	GPIO11 引脚可外接
步骤	1. 根据用户配置进行选用 XO 或者 RCO 作为 PLL 时钟源进行初始化时钟。 2. 检查时钟初始化状态进行判断使用的时钟源。 3. 设置将时钟分频后，输出到 GPIO11。

©2026 旋智电子科技（上海）有限公司

栏目	描述
预期结果	1. 运行程序后打印正常。 2. GPIO11 引脚输出分频后的时钟波形。
注意点	

3 GPIO

3.1 GPIO 电平翻转检测

栏目	描述
工程名	GPIO_Edge_Detect
功能	采用边沿检测方式，判断检测 GPIO 是否发生电平反转。
硬件环境	1. 芯片 DEMO_GPIO_OUTPUT 、 DEMO_GPIO_INPUT 可外接，并将 DEMO_GPIO_OUTPUT 与 DEMO_GPIO_INPUT 连接。
步骤	1. 配置 DEMO_GPIO_OUTPUT 为输出。 2. 配置 DEMO_GPIO_INPUT 为输入，并配置抗尖峰脉冲滤波。 3. 配置 DEMO_GPIO_INPUT 上升沿检测，使能边沿中断，填写 GPIO 边沿中断服务函数，在函数中查询是否存在 DEMO_GPIO_INPUT 上升沿检测标志。 4. 通过 DEMO_GPIO_OUTPUT 翻转输出，为 DEMO_GPIO_INPUT 产生待检测的上升沿。
预期结果	1. 程序运行后，触发检测上升沿中断，打印中断次数持续增加。
注意点	

3.2 GPIO 电平高低检测

栏目	描述
工程名	GPIO_Test_Level
功能	检测 GPIO 电平为高还是低
硬件环境	1. 芯片 DEMO_GPIO_OUTPUT、DEMO_GPIO_INPUT 可外接，并将 DEMO_GPIO_OUTPUT 与 DEMO_GPIO_INPUT 连接。
步骤	1. 配置 DEMO_GPIO_OUTPUT 为输出。 2. 配置 DEMO_GPIO_INPUT 为输入。 3. 每次 DEMO_GPIO_OUTPUT 输出翻转后，使用 DEMO_GPIO_OUTPUT 进行回采，并使用 DEMO_GPIO_INPUT 进行输入电平采集。
预期结果	1. 程序运行后，循环打印 DEMO_GPIO_OUTPUT 的回采电平与 DEMO_GPIO_INPUT 的采集电平。
注意点	

©2026 旋智电子科技（上海）有限公司

4 CRC

4.1 CRC 正常工作检测

栏目	描述
工程名	CRC_Calculate
功能	配置 CRC 并计算一段字符串，将计算结果与网上计算过的结果对比，以检查 CRC 是否工作正常
硬件环境	
步骤	1. 配置 CRC 计算的模式对输入的字符串进行 CRC 计算，并对计算的结果和网站计算结果进行比较。
预期结果	1. CRC 计数的结果与网站计算的结果一致。
注意点	

5 timer

5.1 Timer 事件计数器

栏目	描述
工程名	TIMER_Event_Counter
功能	外部输入引脚当作计数事件，触发 TIMER 超时
硬件环境	1. 芯片 DEMO_GPIO_OUTPUT、DEMO_GPIO_INPUT 可外接，并将 GPIO28 与 DEMO_GPIO_INPUT 连接。
步骤	1. 配置 DEMO_GPIO_OUTPUT 为输出。 2. 配置 DEMO_GPIO_INPUT 为输入。 3. 初始化 Timer，配置 DEMO_GPIO_INPUT 为外部输入引脚，计数模式为事件计数器并重新配置事件加载值。 4. 使能 timer 中断，填写 timer 中断服务函数，在函数中记录中断触发次数，并清除中断。 5. 使能 timer。 6. 周期性翻转 DEMO_GPIO_OUTPUT 引脚 100 次触发 timer 超时中断。
预期结果	1. DEMO_GPIO_OUTPUT 引脚翻转 100 次触发一次 timer 超时中断。
注意点	

5.2 Timer 门控计数器

栏目	描述
工程名	TIMER_Gated_TIMER
功能	外部输入引脚当作 timer 使能信号

©2026 旋智电子科技（上海）有限公司

栏目	描述
硬件环境	
步骤	1. 配置 DEMO_GPIO_INPUT 为输入。 2. 初始化 timer，配置 DEMO_GPIO_INPUT 为外部输入引脚，高电平有效，并配置 timer 模式为门控定时器。 3. 使能 timer 中断，填写 timer 中断服务函数，在函数中记录中断触发次数，并清除中断。 4. 使能 timer。
预期结果	1. 程序运行后，当 DEMO_GPIO_INPUT 输入高电平时，使能 timer 计数，达到触发事件后触发超时中断，当 DEMO_GPIO_INPUT 输入低电平时，失能 timer 计数。
注意点	

5.3 Timer 触发中断

栏目	描述
工程名	TIMER_Int
功能	设置 Timer 开始计时，结束后进入中断
硬件环境	
步骤	1. 初始化 timer，配置为每 1000ms 触发 Timer 中断。 2. 使能 timer 中断，填写 timer 中断服务函数，在函数中记录中断触发次数，并清除中断。 3. 使能 timer。
预期结果	1. 每 1000ms 打印触发 timer 中断。
注意点	

6 WDT

6.1 WDT 喂狗

栏目	描述
工程名	WDT_Feed_DOG
功能	WTD 计时一定时间，但在结束之前重新喂狗，使其不进入超时中断
硬件环境	
步骤	1. 检查复位事件是否被置位。 2. 初始化 WDT，配置为每 50ms 触发一次中断。 3. 使能 WDT 超时产生复位事件。 4. 使能 WDT 中断，填写中断服务函数。 5. 使能 WDT。

©2026 旋智电子科技（上海）有限公司

栏目	描述
	6. 每 30ms 进行一次喂狗，喂 100 次。
预期结果	1. 停止喂狗时触发复位。
注意点	

6.2 WDT 触发中断

栏目	描述
工程名	WDT0_INT
功能	WTD 计时一定时间，结束后进入中断。
硬件环境	
步骤	1. 初始化 WDT， 配置为每 1000ms 触发一次中断。 2. 使能 WDT 中断，填写中断服务函数，在函数中查询 WDT 超时发生标志。如标志有效，则记录中断触发次数，并清除中断。 3. 使能 WDT。
预期结果	1. 每 1000ms 打印触发 WDT 中断以及打印触发中断次数。
注意点	

7 PWM

7.1 PWM 输出互补波形

栏目	描述
工程名	PWM_Complementary_Pair_Channel
功能	设置某路 PWM，并使其两个通道（A/B）产生一对互补的波形
硬件环境	1. 芯片 DEMO_PWM_A_PIN、DEMO_PWM_B_PIN 引脚可外接。
步骤	1. 配置 DEMO_PWM_A_PIN、DEMO_PWM_B_PIN 为 PWM 的输出引脚。 2. 配置 PWM 为双通道，频率为 20Khz，互补波形死区为 1000ns。 3. 配置 PWM 占空比为 75%。 4. 使能 PWM。
预期结果	1. 通过示波器等检测工具测量 DEMO_PWM_A_PIN、DEMO_PWM_B_PIN，可获得频率为 20Khz、占空比为 75%、死区为 1000ns 的互补波形。
注意点	

7.2 PWM 输出单通道波形

栏目	描述
工程名	PWM_Single_Channel

©2026 旋智电子科技（上海）有限公司

栏目	描述
功能	PWM 输出单通道波形。
硬件环境	芯片 DEMO_PWM_A_PIN 引脚可外接。
步骤	1. 配置 PWM 通道引脚。 2. 配置 PWM 为单通道，频率为 20Khz。 3. 配置 PWM 占空比为 50%。 4. 使能 PWM。
预期结果	通过示波器等检测工具测量 DEMO_PWM_A_PIN，可获得频率为 20Khz、占空比为 50%的 PWM 波形。
注意点	

7.3 PWM 输出三相波形

栏目	描述
工程名	PWM_TripPhase_Waveform
功能	设置 PWM 输出电机三相波形，三相波形无相位差
硬件环境	1. 芯片 PWM_U、PWM_V、PWM_W 引脚可外接。
步骤	1. 配置 PWM_U、PWM_V、PWM_W 的 PWM 引脚。 2. 配置 PWM_U、PWM_V、PWM_W，频率为 20Khz，互补波形死区为 1000ns。 3. 使能 PWM_U、PWM_V、PWM_W。
预期结果	1. 通过示波器等检测工具测量 LO_U、HO_U，可获得频率为 20Khz、占空比为 50%、死区为 1000ns 的互补波形。 2. 通过示波器等检测工具测量 LO_V、HO_V，可获得频率为 20Khz、占空比为 50%、死区为 1000ns 的互补波形。 3. 通过示波器等检测工具测量 LO_W、HO_W，可获得频率为 20Khz、占空比为 50%、死区为 1000ns 的互补波形。
注意点	

7.4 级联同步方式设置 PWM 相位差-首节点软件触发

栏目	描述
工程名	PWM_Software_Group_SYNCI
功能	设置 PWM 输出电机三相波形，使用级联同步方式，通过软件信号施加首节点，产生 120° 相位差。
硬件环境	1. 芯片 PWM_U、PWM_V、PWM_W 引脚可外接。
步骤	1. 配置 PWM_U、PWM_V、PWM_W 输出通道引脚。 2. 配置 PWM_U、PWM_V、PWM_W，频率为 20Khz，互补波形死区为 1000ns。

©2026 旋智电子科技（上海）有限公司

栏目	描述
	3. 配置 PWM_U、PWM_V、PWM_W 占空比为 33.33%。 4. 配置非级联同步，由软件触发，同步后三相相位差 120° 5. 使能 PWM_U、PWM_V、PWM_W。 6. 施加 PWM_U、PWM_V、PWM_W 软件同步。
预期结果	1. 通过示波器等检测工具测量 PWM_U 引脚，可获得频率为 20Khz、占空比为 33.33%、死区为 1000ns 的互补波形。 2. 通过示波器等检测工具测量 PWM_V 引脚，可获得频率为 20Khz、占空比为 33.33%、死区为 1000ns 的互补波形。 3. 通过示波器等检测工具测量 PWM_W 引脚，可获得频率为 20Khz、占空比为 33.33%、死区为 1000ns 的互补波形。 4. 通过示波器等测量工具测量 PWM_U、PWM_V、PWM_W 的各自 A/B 通道的相位差为 120° 。
注意点	

7.5 非级联同步方式设置 PWM 相位差-软件触发

栏目	描述
工程名	PWM_Software_Global_SYNCI
功能	设置 PWM 输出电机三相波形，使用非级联同步方式，通过软件信号，产生 120° 相位差。
硬件环境	1. 芯片 PWM_U、PWM_V、PWM_W 引脚可外接。
步骤	1. 配置 PWM_U、PWM_V、PWM_W 输出通道引脚。 2. 配置 PWM_U、PWM_V、PWM_W，频率为 20Khz，互补波形死区为 1000ns。 3. 配置 PWM_U、PWM_V、PWM_W 占空比为 33.33%。 4. 配置非级联同步，由软件触发，同步后三相相位差 120° 5. 使能 PWM_U、PWM_V、PWM_W。 6. 施加 PWM_U、PWM_V、PWM_W 软件同步。
预期结果	1. 通过示波器等检测工具测量 PWM_U 引脚，可获得频率为 20Khz、占空比为 33.33%、死区为 1000ns 的互补波形。 2. 通过示波器等检测工具测量 PWM_V 引脚，可获得频率为 20Khz、占空比为 33.33%、死区为 1000ns 的互补波形。 3. 通过示波器等检测工具测量 PWM_W 引脚，可获得频率为 20Khz、占空比为 33.33%、死区为 1000ns 的互补波形。 4. 通过示波器等测量工具测量 PWM_U、PWM_V、PWM_W 的各自 A/B 通道的相位差为 120° 。
注意点	

©2026 旋智电子科技（上海）有限公司

7.6 非级联同步方式设置 PWM 相位差-GPIO 触发

栏目	描述
工程名	PWM_GPIO_SYNCI
功能	设置 PWM 输出电机三相波形，使用非级联同步方式，通过 GPIO 高电平触发，产生 120° 相位差。
硬件环境	1. 芯片 DEMO_GPIO_INPUT、PWM_U、PWM_V、PWM_W 引脚可外接。
步骤	1. 配置 DEMO_GPIO_INPUT 为输入模式并使能下拉。 2. 配置 PWM_U、PWM_V、PWM_W 输出通道引脚。 3. 配置 PWM_U、PWM_V、PWM_W，频率为 20Khz，互补波形死区为 1000ns。 4. 配置 PWM_U、PWM_V、PWM_W 占空比为 33.33%。 5. 配置非级联同步，由 GPIO 高电平触发，同步后三相相位差 120° 。 6. 使能 PWM_U、PWM_V、PWM_W。
预期结果	1. 通过示波器等检测工具测量 PWM_U 引脚，可获得频率为 20Khz、占空比为 33.33%、死区为 1000ns 的互补波形。 2. 通过示波器等检测工具测量 PWM_V 引脚，可获得频率为 20Khz、占空比为 33.33%、死区为 1000ns 的互补波形。 3. 通过示波器等检测工具测量 PWM_W 引脚，可获得频率为 20Khz、占空比为 33.33%、死区为 1000ns 的互补波形。 4. 通过示波器等测量工具测量 PWM_U、PWM_V、PWM_W 的各自 A/B 通道的相位差为 120° 。
注意点	

7.7 非级联同步方式设置 PWM 相位差-Timer 触发

栏目	描述
工程名	PWM_Timer_SYNCI
功能	设置 PWM 输出电机三相波形，使用非级联同步方式，通过 Timer 计时触发，产生 120° 相位差。
硬件环境	芯片 PWM_U、PWM_V、PWM_W 引脚可外接。
步骤	1. 配置 TIMER 定时器 1000ms 触发 PWM 同步事件。 2. 配置 PWM_U、PWM_V、PWM_W 输出通道引脚。 3. 配置 PWM_U、PWM_V、PWM_W，频率为 20Khz，互补波形死区为 1000ns。 4. 配置 PWM_U、PWM_V、PWM_W 占空比为 33.33%。 5. 配置非级联同步，由软件触发，同步后三相相位差 120° 6. 使能 PWM_U、PWM_V、PWM_W，和 TIMER 定时器。
预期结果	1. 通过示波器等检测工具测量 PWM_U 引脚，可获得频率为 20Khz、占空比为 33.33%、死区为 1000ns 的互补波形。

©2026 旋智电子科技（上海）有限公司

栏目	描述
	2. 通过示波器等检测工具测量 PWM_V 引脚，可获得频率为 20Khz、占空比为 33.33%、死区为 1000ns 的互补波形。 3. 通过示波器等检测工具测量 PWM_W 引脚，可获得频率为 20Khz、占空比为 33.33%、死区为 1000ns 的互补波形。 4. 通过示波器等测量工具测量 PWM_U、PWM_V、PWM_W 的各自 A/B 通道的相位差为 120°。
注意点	

7.8 PWM 触发 ADC 采集

栏目	描述
工程名	PWM_Request_ADC_and_Observation
功能	设置 PWM 时基为 0 时，触发 ADC 采集。
硬件环境	1. 芯片 DEMO_PWM_A_PIN、DEMO_PWM_B_PIN 引脚可外接。
步骤	1. 初始化 ADC，并配置 ADC 平均值次数、采集通道、采集引脚并使能 ADC。 2. 配置 PWM 通道的 GPIO 引脚。 3. 配置 PWM 为双通道，频率为 2Khz，互补波形死区为 100ns。 4. 配置 PWM 占空比为 25%。 5. 配置每次 PWM 时基为 0 时，触发 ADC 采集。 6. 填写 ADC 的 IRQ 中断服务函数，使能 ADC 中断，使能中断。 7. 使能 ADC，并运行 PWM。
预期结果	1. 通过示波器等检测工具测量 DEMO_PWM_A_PIN、DEMO_PWM_B_PIN，可获得频率为 2Khz、占空比为 25%、死区为 100ns 的互补波形。 2. 打印 ADC 采集的电压。
注意点	

7.9 PWM 信号封锁-GPIO 触发

栏目	描述
工程名	PWM_GPIO_Trigger_TZ
功能	1. 设置 GPIO 为 TZ0 的触发源，并设置为持续封锁模式，当电平满足触发要求时，触发 TZ 事件，关闭 PWM 输出，直到清除 TZ 标志位。 2. 另一组 GPIO 分别作为 TZ1/2 触发源，并设置为单次封锁模式，当电平满足触发要求时，触发 TZ 事件，关闭当周期 PWM 输出。
硬件环境	芯片 DEMO_PWM_A_PIN、DEMO_TZ0_PIN、DEMO_TZ1_PIN、DEMO_TZ2_PIN 引脚可外接。
步骤	1. 配置 DEMO_TZ0_PIN、DEMO_TZ1_PIN、DEMO_TZ2_PIN 为 GPIO 引脚，并使能滤波。 2. 配置 DEMO_PWM_A_PIN 为 PWM 引脚。

©2026 旋智电子科技（上海）有限公司

栏目	描述
	3. 设置 PWM 单通道，频率为 20Khz ，占空比为 50%。 4. 配置当 DEMO_TZ1_PIN、DEMO_TZ2_PIN 为低电平时，PWM 触发持续封锁，GPIO28 为低电平时，PWM 触发单次封锁，输出高阻态。 5. 使能持续封锁和单次封锁中断，编写中断服务函数。 6. 使能 PWM。
预期结果	1. DEMO_TZ0_PIN、DEMO_TZ1_PIN、DEMO_TZ2_PIN 为高时，通过示波器等检测工具测量 DEMO_PWM_A_PIN，可获得频率为 20Khz、占空比为 25%的波形。 2. 当 DEMO_TZ0_PIN 为低电平时，通过示波器等检测工具测量 DEMO_PWM_A_PIN，每 500ms 触发一次，PWM 单次封锁。 3. 当 DEMO_TZ1_PIN、DEMO_TZ2_PIN 为低电平时，通过示波器等检测工具测量 DEMO_PWM_A_PIN，PWM 触发持续封锁。
注意点	

7.10 PWM 信号封锁-过压触发

栏目	描述
工程名	PWM_Trigger_TZ_By_Over_Voltage
功能	采用 Comparator 对 IO 口进行过流或欠流检测，最后设置 Comparator 的信号作为 PWM 的 TZ 触发源。设置为单次封锁模式，若过流或欠流发生时，关闭当周期 PWM 输出。
硬件环境	芯片 GPIO1、DEMO_PWM_A_PIN、DEMO_PWM_B_PIN 引脚可外接。
步骤	1. 配置 PWM 为双通道，频率为 20Khz，互补波形死区为 1000ns 波形。 2. 配置 PWM 波形的占空比为 25%。 3. 配置触发过流/欠流事件时，PWM 触发单次信号封锁，信号封锁后，PWM 输出为低，并配置 COMP，过压为 2000mV，欠压为 1000mV，过滤时间为 300ns。 4. 填写 PWM 触发信号封锁的 IRQ 中断服务函数，使能中断。 5. 使能运行 PWM。
预期结果	1. 向 GPIO1 施加 1200mV 电压，通过示波器等检测工具测量 DEMO_PWM_A_PIN、DEMO_PWM_B_PIN，可获得频率为 20Khz、占空比为 25%、死区为 1000ns 的互补波形。 2. 改变 GPIO1 施加电压，大于 1400mV，通过示波器等检测工具测量 DEMO_PWM_A_PIN、DEMO_PWM_B_PIN，每 1s 触发一次，PWM 单次封锁。 3. 还原步骤 1，现象不变。 4. 改变 GPIO0 施加电压，小于 1000mV，通过示波器等检测工具测量 DEMO_PWM_A_PIN、DEMO_PWM_B_PIN，每 1s 触发一次，PWM 单次封锁。 5. 还原步骤 1，现象不变。

©2026 旋智电子科技（上海）有限公司

栏目	描述
注意点	调整施加 GPIO0 电压时，务必进行微调，高出 1400mV 即可，防止烧毁芯片。

7.11 PWM 信号封锁-COMP 触发两路 TZ

栏目	描述
工程名	PWM_Trigger_TZ_Separately_By_COMP
功能	检查 ADC 的 GPIO1 电压，判断是否超过阈值，在 PWM 的 DC 子模块里面，发送信号到 TZ 子模块，判断是否关闭 PWM 输出
硬件环境	芯片 GPIO1、DEMO_PWM1_A_PIN、DEMO_PWM2_A_PIN 引脚可外接。
步骤	1. 配置 GPIO1 引脚为 GPIO 引脚。 2. 配置 PWM 输出引脚。 3. 配置 PWM 为双通道，频率为 20Khz，互补波形死区为 1000ns。 4. 配置 PWM 占空比并设置 AQ 行为。 5. 配置 PWM 进行级联同步，设置加载值并使能同步。 6. 配置触发 TZ 事件时，PWM 触发信号封锁后，PWM 输出为低。 7. 填写 PWM 触发信号封锁的 IRQ 中断服务函数，使能中断。 8. 使能运行 COMP，使能运行 PWM。
预期结果	1. 通过示波器等检测工具测量 DEMO_PWM1_A_PIN、DEMO_PWM2_A_PIN，可获得频率为 20Khz、死区为 1000ns 的互补波形。 2. 改变 GPIO1 施加电压，大于 1000mV，通过示波器等检测工具测量 DEMO_PWM1_A_PIN、DEMO_PWM2_A_PIN，触发封锁。
注意点	

8 ECAP

8.1 ECAP APWM 模式

栏目	描述
工程名	ECAP_APWM_Function
功能	设置 ECAP 工作在 APWM 模式，并输出 PWM 波形
硬件环境	芯片 DEMO_APWM_PIN 引脚可外接。
步骤	1. 配置 DEMO_APWM_PIN 作为 APWM 输出引脚。 2. 配置 ECAP 作为 APWM 模式并输出 PWM 频率（10kHz）、PWM 输出极性（高）、PWM 占空比（50%）。 3. 运行 ECAP
预期结果	1. 通过示波器等检测工具测量 DEMO_APWM_PIN 输出配置的 PWM 脉冲。
注意点	

©2026 旋智电子科技（上海）有限公司

8.2 ECAP 单通道持续捕获-Absolute 模式

栏目	描述
工程名	ECAP_Continue_Absolute
功能	设置 ECAP 工作在连续采样的 Absolute 模式，并连续对 PWM 波形进行多次事件采样，以此连续计算 PWM 的频率
硬件环境	
步骤	1. 初始化 PWM 为频率（10khz）、占空比（50%）的单通道输出引脚，并使能 PWM。 2. 初始化 ECAP 为 Capture 模式，配置捕获引脚以及触发 ECAP_CEVT3 事件时，捕获计数器的值复位 3. 配置捕获引脚的滤波。 4. 使能 ECAP_CEVT3 事件时的中断产生，并编写中断服务函数。 5. 运行 ECAP。
预期结果	1. 持续打印测量到的 PWM 的频率
注意点	

8.3 ECAP 单通道持续捕获-Delta 模式

栏目	描述
工程名	ECAP_Countinue_Delta
功能	设置 ECAP 工作在连续采样的 Delta 模式，并连续对 PWM 波形进行多次事件采样，以此连续计算 PWM 的频率
硬件环境	
步骤	1. 初始化 PWM 为频率（10khz）、占空比（50%）的单通道输出引脚，并使能 PWM。 2. 初始化 ECAP 为 Capture 模式，配置捕获引脚以及触发 ECAP_CEVT0~3 事件时，捕获计数器的值复位。 3. 配置捕获引脚的滤波。 4. 使能 ECAP_CEVT3 事件时的中断产生，并编写中断服务函数。 5. 运行 ECAP。
预期结果	1. 持续打印测量到的 PWM 的频率
注意点	

8.4 ECAP 单通道一次性捕获-Delta 模式

栏目	描述
工程名	ECAP_Oneshot_Delta
功能	设置 ECAP 工作在 Oneshot 的 Delta 模式，并连续对 PWM 波形进行多次事件采样，以此连续计算 PWM 的频率

©2026 旋智电子科技（上海）有限公司

栏目	描述
硬件环境	
步骤	1. 初始化 PWM 为频率（10khz）、占空比（50%）的单通道输出引脚，并使能 PWM。 2. 初始化 ECAP 为 Capture 模式，配置捕获引脚以及触发 ECAP_CEVT0~3 事件时，捕获计数器的值复位，并使能 oneshot 模式。 3. 配置捕获引脚的滤波。 4. 使能 ECAP_CEVT3 事件时的中断产生，并编写中断服务函数。 5. 运行 ECAP。
预期结果	1. 打印一次测量到 PWM 的频率
注意点	

8.5 ECAP 单通道一次性捕获- Absolute 模式

栏目	描述
工程名	ECAP_Oneshot_Absolute
功能	设置 ECAP 工作在 oneshot 的 Absolute 模式，并对 PWM 波形进行多次事件采样，以此计算 PWM 的频率
硬件环境	
步骤	1. 初始化 PWM 为频率（10khz）、占空比（50%）的单通道输出引脚，并使能 PWM。 2. 初始化 ECAP 为 Capture 模式，配置捕获引脚以及触发 ECAP_CEVT3 事件时，捕获计数器的值复位，并使能 oneshot 捕获模式。 3. 配置捕获引脚的滤波。 4. 使能 ECAP_CEVT3 事件时的中断产生，并编写中断服务函数。 1. 运行 ECAP。
预期结果	1. 打印一次测量到 PWM 的频率
注意点	

8.6 ECAP 双通道持续捕获-Delta 模式

栏目	描述
工程名	ECAP_Countinue_Absolute_Dual_Pin
功能	使用连续捕获模式，以绝对时间戳计数方式，对 PWM 双通道波形进行频率和相位差计算。
硬件环境	
步骤	1. 初始化 PWM 为频率（10khz）、占空比（50%）、死区 400ns 的互补波形。 2. 初始化 ECAP 为双通道模式，并将 PWM 产生的两路信号送进 ECAP 进行捕获。初始化函数默认设置捕获事件为 CAP0 上升沿捕获、CAP1 上升沿捕获、CAP2 下降沿捕获、CAP3 下降沿捕获；

©2026 旋智电子科技（上海）有限公司

栏目	描述
	3. 配置为 ECAP_CEVT0~3 事件时，捕获计数器的值复位。 4. 使能 ECAP_CEVT3 事件时的中断产生，并编写中断服务函数。 5. 运行 ECAP。
预期结果	1. 持续打印 PWM 的频率和相位差
注意点	

9 ADC

9.1 ADC 单端采样-软件触发

栏目	描述
工程名	ADC_Single_End_Result
功能	设置 ADC 工作在单端采样（有一端被接到 GND）。
硬件环境	芯片 GPIO1 引脚可外接。
步骤	1. 初始化 ADC 通道，P 端配置为 GPIO0，N 端配置为 GND，采样时间、转换时间，设置为软件触发。 2. 使能 ADC。 3. 强制触发 ADC 采样。 4. 超时等待 ADC 采样、转换完成。 5. 读取 ADC 采样 GPIO 上的电压结果，并进行打印。
预期结果	1. 循环打印 ADC 采样 GPIO 的电压值。
注意点	

9.2 ADC 单端采样-定时器触发

栏目	描述
工程名	ADC_Triggered_by_Timer
功能	通过定时器触发 ADC 采样
硬件环境	芯片 GPIO1 引脚可外接
步骤	1. 配置 Timer 触发 ADC 转换时间为每 1000ms 一次。 2. 初始化 ADC 通道，P 端配置为 GPIO0，N 端配置为 GND，配置采样时间、转换时间，设置为 Timer 触发。 3. 使能 ADC 中断，并编写中断服务函数。 4. 使能 ADC 和 TIMER。
预期结果	1. 每秒打印 ADC 采样 GPIO 的电压值。
注意点	

©2026 旋智电子科技（上海）有限公司

9.3 ADC 双端采样-软件触发

栏目	描述
工程名	ADC_Differential_Result
功能	设置 ADC 工作在双端采样。
硬件环境	芯片 GPIO0、GPIO1 引脚可外接。
步骤	1. 初始化 ADC 通道，P 端配置为 GPIO1，N 端配置 GPIO0，配置采样时间、转换时间，设置为软件触发。 2. 使能 ADC。 3. 强制触发 ADC 采样。 4. 超时等待 ADC 采样、转换完成。 5. 循环进行 ADC 采集。
预期结果	1. 循环打印 ADC 双端采样 GPIO 的电压值。
注意点	

9.4 ADC 增益及偏移校准

栏目	描述
工程名	ADC_Trim
功能	校准 ADC 的增益及偏移数值
硬件环境	芯片 GPIO0、GPIO1 引脚可外接。
步骤	1. 初始化 ADC 的 CH0 用于采集 DVSS。 2. 初始化 ADC 的 CH1 用于采集 3.3V（GPIO1 接 3.3V，GPIO0 接 GND）。 3. 根据采集结果计算增益及偏移校准。
预期结果	1. 将 GPIO0 接 3.3V，GPIO1 接 GND 后，启动程序。 2. 打印旧的增益及偏移值后，打印本次测量计算得到的增益及偏移值。
注意点	

9.5 ADC 开路检测

栏目	描述
工程名	ADC_Detect_Open_Circuit
功能	检测采样器通路上被采样的 ADC GPIO 是否存在虚焊等导致开路的情况
硬件环境	
步骤	1. 将 PIN 脚切成 GPIO 功能，设置为输入，并使能 GPIO 下拉，释放 IO 上残存的电量；将 ADC 采样保持的正端强制接地，释放通路上的残存电量； 2. 将 ADC 采样保持的正端恢复断开状态，将 PIN 脚切成 ADC 功能，降低 ADC 模块的时钟频率，设置较长采样时间，以期获得更为准确的测试数据，随后开始测量，获得电压数值 LowResult；

©2026 旋智电子科技（上海）有限公司

栏目	描述
	3. 将 PIN 脚切成 GPIO 功能，设置为输入，并使能 GPIO 上拉，给 IO 的通路进行预充电；将 ADC 采样保持的正端强制拉高，对采样保持器通路进行预充电； 4. 将 ADC 采样保持的正端恢复断开状态，将 PIN 脚切成 ADC 功能，降低 ADC 模块的时钟频率，设置较长采样时间，以期获得更为准确的测试数据，随后开始测量，获得电压数值 HighResult；
预期结果	1. 如果 LowResult 小于 200mV 且 HighResult 大于 3200mV，则打印 ADC 引脚断开。 2. 否则，打印 ADC 引脚连接。
注意点	

9.6 ADC 短路检测

栏目	描述
工程名	ADC_Short_Detect_Circuit
功能	检测采样器通路上被采样的 ADC GPIO 是否存在短路情况
硬件环境	
步骤	1. ADC 测量输入 PIN 脚电压，设为结果 Result1。 2. 将 PIN 脚切成 GPIO 功能，设置为输入，给输入 PIN 接弱下拉 GND，ADC 测量输入 PIN 脚电压，设为结果 Result2。 3. 将 PIN 脚切成 GPIO 功能，设置为输入，给输入 PIN 接弱上拉 VDD，ADC 测量输入 PIN 脚电压，设为结果 Result3。
预期结果	如果 Result1 与 Result2 与 Result3 的差值在 200mV 内，那么输入引脚短路。
注意点	

©2026 旋智电子科技（上海）有限公司

10 TSENSER

10.1 Tsenser 测量 MCU 温度

栏目	描述
工程名	Tsenser
功能	使用芯片内部温度传感器测量芯片结温。
硬件环境	
步骤	1. 调用 ADC_GetTemperature 进行测量芯片的结温。
预期结果	循环打印测量到的芯片结温。
注意点	

11 PGA

11.1 PGA 校准

栏目	描述
工程名	PGA_Trim
功能	校准 PGA
硬件环境	
步骤	1. 配置 PGA 进行 4 倍放大。 2. 初始化 ADC 通道，P 端配置为 PGA 输出的正极，N 端配置为 PGA 输出的负极，配置采样时间、转换时间，设置为软件触发，并使能浮点结果。 3. 配置 ADC 的偏移为 0 以及增益为 32768 4. 使能 ADC。 5. 强制触发 ADC 进行采样。 6. 等待 ADC 转换完成。 7. 获取 ADC 采样的结果。
预期结果	1. 将 PGA 输出的正极和负极接 GND 进行测量 ADC 采样的结果。 2. 然后将 PGA 输出的正极接 0.6V，负端接 GND 进行测量 ADC 采样的结果。 3. 根据两次采样的结果进行计算包含 PGA 校准的 ADC 的偏移和增益。
注意点	

©2026 旋智电子科技（上海）有限公司

12 Comparator

12.1 设置比较器阈值及检测

栏目	描述
工程名	Comparator
功能	设置 Comparator 阈值，检测输入是否超过此阈值
硬件环境	GPIO0、GPIO1、GPIO2 可外接
步骤	1. 配置 COMP 进行检测 GPIO2 电压，检测的高阈值为 1.4V，检测的低阈值为 1V，并设置滤波窗口为 300ns，使能 COMP。 2. 循环查询 COMP 的标志位，检测 GPIO 的电压是否过低或者过高。
预期结果	1. 当向 GPIO2 施加的电压超过 1.4V 时，输出电压过高提示，当施加的电压低于 1.0V 时，输出电压过低提示。 2. 当向 GPIO2 施加的电压超过 1.4V 时，GPIO1 拉高，当施加的电压低于 1.0V 时 GPIO0 拉高。
注意点	

SPINTROL

©2026 旋智电子科技（上海）有限公司

13 DAC

13.1 DAC 正常工作检测

栏目	描述
工程名	DAC
功能	设置 DAC 输出某一数值的电压，并利用 ADC 进行采样，判断 ADC 的结果是否在合理范围区间，以此判断 DAC 是否工作正常
硬件环境	
步骤	1. 配置 ADC 用以采集模拟量值。 2. 配置 DAC 用以产生模拟量值。 3. 使用 DAC 周期性产生 0~3300mV 的模拟量电压值。 4. 使用 ADC 回采 DAC 产生的模拟量电压值。
预期结果	1.周期性打印回采到 DAC 产生的模拟量电压值。
注意点	

14 CRC

14.1 CRC 正常工作检测

栏目	描述
工程名	CRC_Calculate
功能	配置 CRC 并计算一段字符串，将计算结果与网上计算过的结果对比，以检查 CRC 是否工作正常
硬件环境	
步骤	1. 调用 CRC 接口函数，校验不用模式下产生的 CRC 校验码是否正确。
预期结果	1. 所有模式下 CRC 校验码计算正确。
注意点	

15 UART

15.1 UART 自动波特率模式

栏目	描述
工程名	UART_Auto_Baudrate
功能	UART 根据接收到的 0x7F 字节， 自动设置波特率。
硬件环境	

©2026 旋智电子科技（上海）有限公司

栏目	描述
步骤	1. 配置 PIN_MUX 用于 UART 的 TXD 和 RXD，使能 UART 时钟。 2. 设置为自动波特率模式。 3. 等待接收 0x7F 字节，进行自动波特率检测。 4. 自动波特率确定后，检测接收到的字节是否为 0x55，用以确认波特率是否正确。
预期结果	1. 自动波特率设置成功。
注意点	

15.2 UART 阻塞方式-发送接收检查

栏目	描述
工程名	UART_Tx_and_Check_Rx
功能	与“UART_Rx_and_Sent_Back”配套，实现两个设备使用 UART 通信数据。
硬件环境	1. DEMO_GPIO_OUTPUT 引脚可外接。
步骤	1. 配置 PIN_MUX 用于 UART 的 TXD 和 RXD，配置波特率并使能 UART。 2. 发送 1 字节随机数据。 3. 等待接收 1 字节随机数据。 4. 检测发送和接收的数据是否一致。
预期结果	1. 发送和接收检测都成功，通过示波器等观测设备测量 DEMO_GPIO_OUTPUT 引脚，电平为低。
注意点	

15.3 UART 阻塞方式-接收发送

栏目	描述
工程名	UART_Rx_and_Sent_Back
功能	与“UART_Tx_and_Check_Rx”配套，实现两个设备使用 UART 通信数据。
硬件环境	
步骤	1. 配置 PIN_MUX 用于 UART 的 TXD 和 RXD，配置波特率并使能 UART。 2. 等待接收 1 字节随机数据。 3. 发送 1 字节随机数据。
预期结果	
注意点	

15.4 UART 非阻塞中断方式-接收不定长数据

栏目	描述
工程名	UART_Rx_Undefined_Length

©2026 旋智电子科技（上海）有限公司

栏目	描述
功能	使用中断方式，利用接收超时标志位实现不定长数据的接收。
硬件环境	
步骤	1. 配置 PIN_MUX 用于 UART 的 TXD 和 RXD，配置波特率并使能 UART。 2. 配置接收缓冲区中断阈值。 3. 使能 UART 中断。
预期结果	1. 可以接收到不定长数据，并将接收到的数据进行发送。
注意点	

16 I2C

16.1 I2C 主机模式-阻塞方式

栏目	描述
工程名	I2C_Master_Polling_TxRx
功能	I2C 作为 Master，采用阻塞方式，周期性向 Slave 发送和接收数据。
硬件环境	
步骤	5. 配置 PIN_MUX 用于 I2C 的 SDA 和 SCK。 6. 配置 I2C 为主机模式，并设置通信速率。 7. 使能 I2C。 8. 阻塞方式向从机设备发送数据。 9. 阻塞方式接收从机设备的数据。 10. 检测发送和接收的数据是否一致。
预期结果	2. 串口打印发送和接收的数据一致。
注意点	

16.2 I2C 从机模式-阻塞方式

栏目	描述
工程名	I2C_Slave_Polling_TxRx
功能	与“I2C_Master_Polling_TxRx”配套，作为 Slave，接收 Master 的数据，并将其发送回 Master。
硬件环境	
步骤	4. 配置 PIN_MUX 用于 I2C 的 SDA 和 SCK。 5. 配置 I2C 为从机模式，并设置从机地址和通信速率。 6. 使能 I2C。 7. 阻塞方式接收主机设备数据。 8. 阻塞方式将接收到的数据，发送回主机设备。

©2026 旋智电子科技（上海）有限公司

栏目	描述
预期结果	2. 串口打印接收到主机设备的字节数，并发送回主机设备。
注意点	

16.3 I2C 主机模式-非阻塞中断方式

栏目	描述
工程名	I2C_Master_Int_TxRx
功能	设置 I2C 作为 Master，采用中断方式向 Slave 发送数据，并采用中断方式接收 Slave 返回的数据，对比发送和接收的数据是否一致。
硬件环境	
步骤	9. 配置 PIN_MUX 用于 I2C 的 SDA 和 SCK。 10. 配置 I2C 为主机模式，并设置通信速率。 11. 使能 I2C NVIC 中断。 12. 使能 I2C。 13. 中断方式向从机设备发送数据。 14. 中断方式接收从机设备数据。 15. 检测发送和接收的数据是否一致。
预期结果	1. 串口打印发送和接收的数据一致。
注意点	

16.4 I2C 从机模式-非阻塞中断方式

栏目	描述
工程名	I2C_Slave_Int_TxRx
功能	与“I2C_Master_INT_TxRx”配套，作为 Slave，接收 Master 的数据，并将其发送回 Master。
硬件环境	
步骤	16. 配置 PIN_MUX 用于 I2C 的 SDA 和 SCK。 17. 配置 I2C 为从机模式，并设置从机地址和通信速率。 18. 使能 I2C NVIC 中断。 19. 使能 I2C。 20. 中断方式接收主机设备数据。 21. 检查接收到的主机设备数据，CRC 校验码是否正确。 22. 如 CRC 校验码正确，则将接收到的数据，通过中断方式发送回主机设备。
预期结果	2. 串口打印接收到主机设备数据，CRC 校验码正确。
注意点	

©2026 旋智电子科技（上海）有限公司

17 SPI

17.1 SPI 主机模式-阻塞方式

栏目	描述
工程名	SPI_Master_Polling_TxRx
功能	I2C 作为 Master，采用阻塞方式，向 Slave 发送和接收数据。
硬件环境	
步骤	1. 配置 PIN_MUX 用于 CS、CLK、MISO 和 MSIO。 2. 配置 SPI 为主机模式，并设置数据宽度和通信速率。 3. 使能 SPI。 4. 阻塞全双工方式向从机设备发送数据，并检测接收的数据与发送的数据是否一致。
预期结果	1. 串口打印操作成功。
注意点	

17.2 SPI 从机模式-阻塞方式

栏目	描述
工程名	SPI_Slave_Polling_TxRx
功能	与“SPI_Master_Polling_TxRx”配套，向 Master 发送和接收数据。
硬件环境	
步骤	1. 配置 PIN_MUX 用于 CS、CLK、MISO 和 MSIO。 2. 配置 PIN 为从机模式，并设置数据宽度和通信速率。 3. 使能 SPI。 4. 阻塞全双工方式向主机设备发送数据，并检测接收的数据与发送的数据是否一致。
预期结果	1. 串口打印操作成功。
注意点	

17.3 SPI 主机模式-非阻塞中断方式

栏目	描述
工程名	SPI_Master_Int_TxRx
功能	设置 SPI 作为 Master，采用中断方式向 Slave 发送数据，并采用中断方式接收 Slave 返回的数据，对比发送和接收的数据是否一致。
硬件环境	
步骤	1. 配置 PIN_MUX 用于 CS、CLK、MISO 和 MSIO。 2. 配置 PIN 为主机模式，并设置数据宽度和通信速率。 3. 使能 SPI NVIC 中断

©2026 旋智电子科技（上海）有限公司

栏目	描述
	4. 使能 SPI。 5. 通过阻塞方式发送，“数据发送”命令。 6. 中断方式向从机设备发送数据。 7. 通过阻塞方式发送，“数据接收”命令 8. 中断方式接收从机设备的数据。 9. 比较发送的数据与接收的数据是否一致。
预期结果	1. 串口打印发送与接收数据一致。
注意点	

17.4 SPI 从机模式-非阻塞中断方式

栏目	描述
工程名	SPI _Slave_TxRx_INT
功能	与“SSP_Master_TxRx_INT”配套，作为 Slave，接收 Master 的数据，并将其发送回 Master。
硬件环境	
步骤	1. 配置 PIN_MUX 用于 CS、CLK、MISO 和 MSIO。 2. 配置 PIN 为主机模式，并设置数据宽度和通信速率。 3. 使能 SPI NVIC 中断 4. 使能 SPI。 5. 接收到主机发送命令后，通过中断方式进行数据接收。 6. 校验接收到数据 CRC 是否正确。 7. 接收到主机接收命令后，通过中断方式将接收到数据发回主机。
预期结果	1. 串口打印接收到主机设备数据，CRC 校验码正确。
注意点	

©2026 旋智电子科技（上海）有限公司

SPINTROL

18 FLASH

18.1 FLASH 模拟 EEPROM

栏目	描述
工程名	Flash_EEPROM_Emulation_Controller
功能	演示如何使用“EEPROM 算法”将 Flash 模拟成 EEPROM，进行读写测试。
硬件环境	
步骤	1. 调用 EEPROM_Init()进行初始化； 2. 调用 EEPROM_WriteWord()写入随机数据； 3. 调用 EEPROM_ReadWord()读出数据； 4. 比较写入与读出数据是否一致。
预期结果	1. 第一次运行程序时，打印初始化失败，并打印格式化成功。 2. 后续上电只打印初始化成功。 3. 打印测试成功。
注意点	1. 算法实现细节可参考《RC-032_SPC1169 模拟 E2PROM 使用指南》

18.2 FLASH 读写擦操作

栏目	描述
工程名	Flash_Operation
功能	演示 flash 的读写擦操作
硬件环境	
步骤	1. 初始化 FLASH。 2. 对测试扇区进行读写擦测试。
预期结果	1. 擦除、写入以及回读校验成功。
注意点	

18.3 扇区写/擦保护

栏目	描述
工程名	Flash_Sector_Protect
功能	设置 flash 的某一扇区（sector）为保护模式，对此扇区的写入及擦除操作将失败
硬件环境	
步骤	1. 初始化 FLASH。 2. 对测试扇区进行读写擦测试。

©2026 旋智电子科技（上海）有限公司

栏目	描述
	3. 对测试扇区进行写保护设置。 4. 对测试扇区进行读写擦测试。
预期结果	1. 设置写保护之前，打印擦除、写入以及回读校验成功。 2. 设置写保护之后，打印擦除、写入以及回读校验失败。
注意点	

18.4 在 RAM 中运行代码及数据

栏目	描述
工程名	Flash_With_ISR_In_RAM
功能	此示例代码演示，如何新建一个工程，将所有的代码及数据全部放在 RAM 中运行（包括中断向量表）。
硬件环境	
步骤	1. 初始化 FLASH。 2. 将中断向量表拷贝到 RAM 区域。 3. 重映射中断向量表。 4. 初始化定时器和定时器中断。 5. 使能定时器。
预期结果	1. 循环打印 Timer 中断服务函数处理结果
注意点	

19 BUCK

19.1 BUCK 抖频模式

栏目	描述
工程名	BUCK_Dithering
功能	使能 Buck 的 Dithering 模式（抖频模式）
硬件环境	
步骤	1. HV 模块初始化。 2. 设置 HV 模块为 PWM 模式。 3. 启用扩频功能并设置扩频幅度。
预期结果	1. 抖频模式设置成功
注意点	

©2026 旋智电子科技（上海）有限公司

19.2 BUCK 过压检测

栏目	描述
工程名	BUCK_Over_Voltage_Trigger_TZ
功能	配置 Buck 过压检测
硬件环境	
步骤	1. HV 模块初始化。 2. 设置 HV 过压事件。
预期结果	1. 过压时触发过压事件。 2. 过压消失时，过压事件消失。
注意点	

19.3 BUCK 预驱动器

栏目	描述
工程名	High_Voltage_PRE_Driver
功能	使用预驱动器输出三相 PWM。
硬件环境	
步骤	1. HV 模块初始化。 2. 配置 PWM 输出三个互补波形。 3. 使能 PWM。
预期结果	1. 通过预驱动器输出三相 PWM。
注意点	

SPINTROL

©2026 旋智电子科技（上海）有限公司