文档编号：AN2052

上海东软载波微电子有限公司

应用笔记

ES32M0502

修订历史

版本	修改日期	更改概要
V1.0	2025-03-20	初版

地　址：中国上海市徐汇区古美路 1515 号凤凰园 12 号楼 3 楼
E-mail：support@essemi.com
电　话：+86-21-60910333
传　真：+86-21-60914991
网　址：http://www.essemi.com/

目录

第1章 概述

1.1 开发环境

推荐用户使用 Keil5、IAR8.11 进行固件开发。由于 Keil4 不支持 PACK 机制，故不推荐用户使用 Keil4。

1.2 库函数选择

ES32 系列芯片提供 2 种类型库函数 ALD 和 MD：

ALD

提供较为完善的封装，提供更为人性化的 API，适合大部分使用者。

MD

基本上只提供寄存器位域级别的“读”、“写”接口，适合对硬件底层架构较为熟悉的使用者。

如果用户对速度不是要求非常严格，一般情况下推荐使用者使用 ALD 库。可以减少用户学习时间，增加代码可移植性，最终缩短用户产品的开发周期。

1.3 寄存器写保护

为了防止程序异常导致系统出现错误，芯片提供了写保护寄存器，用于防止对被保护的寄存器误操作。FC、IWDT、GPIO 等模块都支持寄存器写保护功能，用户在写入被保护寄存器之前需要先解除写保护状态(允许写)，否则无法对该寄存器进行写入。写入操作完成后，再使能写保护(禁止写入)。在库函数中，都提供了相应的宏定义，方便使用者进行解除保护和使能保护的操作。

1.3.1 FC 写保护

闪存控制器 FC 提供 FC_CMD 和 FC_CTL 寄存器，用于闪存的写访问操作，这些操作是受保护的，即用户无法直接编程或擦除闪存，以避免意外修改闪存数据。

如果用户需要对闪存执行编程或擦除，必须先进行解锁流程。

- 解锁流程需要连续输入 2 组解锁密钥，否则解锁失效，寄存器保持上锁状态。
- 解锁成功后，用户可以通过 FC_STA.CMDULK 位来判定是否成功解锁。
- 当此位状态为 1 时，表示成功解锁，可以进行寄存器写访问操作。

为了保护闪存数据的安全，用户在执行闪存编程或擦除操作前，应该严格遵守解锁流程，避免误操作导致数据丢失。库函数中提供了相应的宏定义，方便用户进行解锁和锁定操作。

解锁流程如下所示：

1. 检查 FC_STA.CMDULK 为 0，确认目前处于锁定状态。
2. 将第一组密钥 0x00112233 写入 FC_UL 寄存器。
3. 将第二组密钥 0x55667788 写入 FC_UL 寄存器。
4. 检查 FC_STA.CMDULK 为 1，确认解锁成功。

进行闪存编程与擦除操作后，建议重新启用锁定功能以保护闪存数据不被意外擦除或覆盖。要执

行此操作，需使用 FC_UL 寄存器填入非密钥值(例如 0x00000000) 以重新上锁闪存控制器。

1. 3. 2 IWDT写保护

IWDT 寄存器、IWDT_BKPR、IWDT_BKRLR 和 IWDT_BKWINR 都是受保护的写访问操作。

解锁方式是对 IWDT_BKKR 写入 0x00005555 解锁码，即可解除写保护功能。如果对 IWDT_BKKR 写入非密钥值，则会重新启用寄存器写访问保护机制。

如要读取 IWDT_BKPR[2:0]、IWDT_BKRLR[11:0] 和 IWDT_BKWINR[11:0] 寄存器，必须先通过将 IWDT_BKPR[15] 、 IWDT_BKRLR[15] 和 IWDT_BKWINR[15] 写 1 并且等待 IWDT_BKFR.BUSY 位清 0 才能读取计数值。

写入寄存器需等待 5 个 PCLK 时钟进行更新，可以读取 IWDT_BKFR.BUSY 位以判断寄存器是否已完成更新。

1. 3. 3 GPIO写保护

通过 GPIOx_LCK 寄存器，可锁定 GPIO 寄存器的写访问功能，包括 GPIOx_MOD、GPIOx_OT、GPIOx_PUD、GPIOx_AFL 和 GPIOx_AFH 寄存器。

锁定的过程需要进行 IO 锁定流程，需对 GPIOx_LCK 寄存器进行 2 次相同数值的写入，每次写入需要 32 位数值，并且 GPIOx_LCK[31:16]必须是 GPIOx_LCK[15:0]的取反值。写入完成后，可读取 GPIOx_LCK.LCKK 位来判断锁定流程是否正确开启了 IO 锁定功能，当 GPIO 寄存器被锁定后，软件无法取消锁定，只能通过系统复位来清除锁定。

1. 3. 4 CMP 写保护

完成 CMP 配置后，您可以将 CMP_CFG1.LOCK 位设置为 1，这样可以使得 CMP_CFG1 和 CMP_CFG2 寄存器都变成只能进行读取操作，包括 LOCK 位。当 CMP 寄存器被锁定后，软件无法取消锁定，只能通过系统复位来清除锁定。

1. 4 写 1 清零寄存器

中断标志寄存器都采用"写 1 清零(C_W1)"的方式来操作，这意味着只有在写 1 时，对应的位才会被清除为 0。对于"写 1 清零(C_W1)"的寄存器，不建议使用"读-修改-写"的方式进行操作，因为容易导致标志误清，进而导致中断丢失的问题。建议在中断处理程序中，使用专门的写操作来清除相应的标志位，确保中断能够正确触发并被处理。

第2章 系统控制

2.1 系统时钟 RCU

系统时钟(SYSCLK)提供以下来源：

- HRC (High Speed Internal RC Oscillator) - 内部高速 16 MHz RC 振荡器
- HOSC (High Speed External Oscillator) - 外部高速时钟振荡器
- PLL (Phase Locked Loop) - 锁相环
- LRC (Low Speed Internal RC Oscillator) - 内部低速 32 kHz RC 振荡器

系统重置后，默认使用内部 16 MHz 高速 RC 振荡器(HRC)作为系统时钟。通过 RCU_CFG.SW 寄存器，可切换不同的系统时钟源。在切换系统时钟源之前，必须确保目标时钟源已经开启并且时钟已经稳定，才可进行更换系统时钟。

读取 RCU_CFG.SWS 寄存器，可确认当前系统时钟源是否已更换完成。需要注意的是，无法通过 RCU_CON 或 RCU_LCON 寄存器来关闭当前的系统时钟源。当误操作时，RCU_CON 或 RCU_LCON 寄存器也不会有任何反应。此外，当 PLL 用于系统时钟源时，PLL 所采用的参考时钟(HRC、HOSC)也无法通过 RCU_CON 或 RCU_LCON 寄存器关闭。

当时钟源不使用时，可以独立设置其开启或关闭，以优化系统功耗。此外，该系统还提供了分频电路，依据应用场景与功耗需求，配置 AHB 与 APB 操作频率。AHB 与 APB 最高可配置为 72MHz。所有周边外设时钟状态都依据所属的总线(Bus)时钟而定。 如 AHB 总线时钟为 HCLK，APB 总线时钟为 PCLK。以下列出使用特殊时钟源的区块：

- STCLK 时钟
- I2CCLK 时钟
- IWDTCLK 时钟
- ADCCLK 时钟

2.1.1 HOSC时钟(HOSCCLK)

外部时钟源 (HOSC Bypass)

使用外部有效的时钟源，需先设定 RCU_CON 缓存器中的 HOSCBYP 和 HOSCON 启用 HOSC Bypass 功能。接着，外部时钟源必须输入至 HOSCI 引脚。当外部时钟源被启用后，内部的 HOSC 将停止振荡，系统将依据外部时钟源提供的频率运行。

外部石英晶体振荡器 (HOSC Crystal)

HOSC 时钟信号是由外部高速晶体振荡器产生。配置 RCU_CON.HOSCON 位控制 HOSC 开启与关闭；HOSC 时钟状态可通过 RCU_CON.HOSCRDY 标志位确认，当 HOSC 时钟稳定时，标志位将被硬件自动配置为高电平。

2.1.2 HRC时钟(HRCCLK)

HRC 时钟信号是由内部高速 16 MHz RC 振荡器产生。通过配置 RCU_CON.HRCON 位，可以控制 HRC 开启或关闭；而 HRC 时钟状态可以通过检查 RCU_CON.HRCRDY 标志位来确认。当

HRC 时钟稳定时，标志位将被硬件自动配置为高电平。

2. 1. 3 PLL时钟(PLLCLK)

锁相回路(Phase-Locked Loop, PLL)可通过 RCU_CFG.PLLSRC 选择 HRC 或 HOSC 作为参考时钟频率， 并且通过 RCU_CFG1.PLLFREQ 预分频将频率分频到适当范围。建议输入时钟频率维持在 4 MHz 左右。

输出时钟频率范围介于 4 至 72 MHz 之间，并由压控振荡器 (Voltage Controlled Oscillator，VCO) 提供。输出时钟频率是 VCO 频率除以一个可变倍率，该倍率可以是整数或非整数。倍率由 RCU_CFG1.FN 与 RCU_CFG1.FK 控制，其中 FN 表示整数倍率，FK 表示小数倍率。根据输出时钟频率区间调整可以设定 RCU_CFG1.FM，FM 可以选择 2、4、6、8、12、16、24、32 或 48。

$$\int VCO = \int PLLIN \times \left(FN + \frac{FK}{2^{19}}\right), 192MHz \leq \int VCO \leq 384MHz$$

$$\int PLL = \frac{\int vco}{FM},\ FM = 2,4,6,8,16,24,32,48$$

使用 RCU_CON.PLLON 位来控制 PLL 开启或关闭；可使用 PLLRDY 标志检查 PLL 时钟的状态，当 PLL 时钟稳定时，标志位将被硬件自动配置为高电平。

PLL 的配置(包括输入时钟源的选择、预分频和倍率设定)必须在 PLL 开启之前完成，开启 PLL 后便无法修改配置，必须关闭 PLL 才可修改配置。修改 PLL 配置的流程如下:

- 关闭 PLL，设定 RCU_CON.PLLON=0
- 确认 RCU_CON.PLLRDY=0 后，PLL 才完成关闭的流程
- 修改 PLL 配置，包括输入时钟源的选择、预分频和倍率设定
- 开启 PLL，通过 RCU_CON.PLLON 位控制 PLL 开启
- 等待 RCU_CON.PLLRDY=1，等待 PLL 输出时钟稳定

2. 1. 4 LRC时钟(LRCCLK)

内部低速 LRC 振荡器，是一个低功耗时钟源，并可以在低功耗模式下让外设独立看门狗(IWDT)和实时时钟(RTC)持续运行；时钟频率大约 32kHz。配置 RCU_LCON.LRCON 位控制 LRC 开启与关闭。LRC 时钟状态可通过 RCU_LCON.LRCRDY 标志位确认，当 LRC 时钟稳定时标志位将被硬件自动配置为高电平。

2. 1. 5 时钟安全系统(CSS)

当系统时钟选择外部石英晶体振荡器时，时钟安全系统(Clock Security System，CSS)是一个重要的系统安全机制。当系统时钟选用 HOSC 或是 PLL 使用 HOSC 倍频时，为了避免 HOSC 发生故障导致系统停机或陷入不安全的状态，可以通过启用或关闭 RCU_CON.HOSCCSSON 来启动或停用 HOSC 时钟安全系统。系统只有在 HOSC 时钟开启且信号稳定时才能启用时钟安全系统，并且在侦测到 HOSC 停止时会自动关闭时钟安全系统并切换成 HRC。

2. 1. 6 特殊区块IP

以下这些特殊区块只能支持特定时钟源

IP \ Source	SYS	AHB	APB	HRC	HOSC	PLL	LRC
ADC 时钟(ADCCLK)	⦿	⦿				⦿	
I2C 时钟(I2CCLK)	⦿		⦿	⦿			
IWDT 时钟(IWDTCLK)							⦿
System Tick 时钟(STCLK)		⦿					

表 2-1 Clock source

2. 1. 7 MCO 时钟 (MCOCLK)

微控制器提供时钟输出功能，允许用户将时钟信号输出到 MCO，不仅可以用于检查，还可当作其他装置或模块的输入时钟源。用户可通过配置 RCU_CFG.MSW 来选择输出时钟，并根据需求设定 RCU_CFG.MPRE 进行分频，可选择 2、4、8、16、32、64 或 128 倍的分频倍率。使用者可以自由选择适合的时钟输出。

2. 2 低功耗唤醒流程

注意事项 1：从低功耗模式唤醒后，必须检查/清除唤醒标志。

当系统从低功耗模式(STOP)通过唤醒事件或引脚 WKUPx 唤醒后，必须清除唤醒标志 SYSCFG_WKSR.WKCLR，并检查唤醒标志 SYSCFG_WKSR.FLAG 是否已清除。

注意事项 2：从低功耗模式(STOP)唤醒后，需检查系统时钟来源是否稳定。

低功耗模式(STOP)唤醒后，系统时钟配置保持睡前状态，但为了确保时钟稳定，请务必确认时钟来源的 RDY 标志；可由 RCU_CFG.SWS 取得当前的时钟源，再判定相对应的稳定标志 RDY。

2. 3 FLASH模块

注意事项 1：FLASH 读保护(RP)配置字

- 配置保护等级为 Lv0 等级，默认不保护。可修改保护等级提升至 Lv1 或 Lv2。在提升保护等级的过程中，程序区内所储存的信息仍会保留。
- 配置保护等级为 Lv1 保护，无法进行程序下载以及数据读出。可修改保护等级，将保护等级降回到 Lv0 保护，这个过程会触发程序区全清除。将保护等级提升至 Lv2 保护，程序区所储存的信息仍然会保留。
- 配置保护等级为 Lv2 保护，将无法进行程序下载、数据读出以及调试，不可修改保护等级。

注意事项 2：FLASH 等待时钟数(Wait Cycle)

系统时钟频率变化时，必须确保有足够的 FLASH 等待周期 FC_CTL.WAIT，支持最多 3 个等待周期，如下表。

系统时钟	FC_CTL.WAIT
72Mhz≥ f_{SYSCLK} > 48Mhz	2
48Mhz≥ f_{SYSCLK} > 24Mhz	1
24Mhz≥ f_{SYSCLK}	0(无等待)

表 2-2 FLASH 等待时钟数(Wait Cycle)

注意事项 3：在进行 FLASH 编程时，无论是否编写相同的数据，在 FLASH 编程前均必须先进行擦除。禁止通过对一个 word 的多次写入实现按 byte 或按 bit 修改。

2.4 配置字

注意事项 1: 配置字修改与重置

当使用 ESLink II / ESLink II Pro，修改芯片配置字之后，需要断电后重新上电，才能正常调试。

注意事项 2: BOR 配置字

BOR 电压等级(Level)，请参考 ES32M0502_Datasheet_C.pdf

注意事项 3: IWDT 配置字

IWDT 配置字开启后，软件无法关闭，预设 1 秒发生 IWDT 复位，支持软件修改复位时间。

第3章 外设

3.1 GPIO 模块

注意事项 1: 未使用的 GPIO 端口处理

在系统中未使用的 GPIO 引脚，建议设置为输出固定电平并悬空，若设置为输入，须加上拉或下拉电阻接到电源或地。

注意事项 2: GPIO 端口复用流程

当需要设置 IO 复用功能时，需要先配置 GPIOx_AFx 寄存器，选择 AFx 复用种类，再将 GPIOx_MOD 配置为复用功能模式(10b)。

注意事项 3: 低功耗模式唤醒引脚

当系统进入低功耗模式(STOP)时，PB0 至 PB8 可以用作唤醒引脚。在设置 IO 复用功能时，需选择 AFx 复用种类为 WKUPx。此外，需开启唤醒功能 SYSCFG_WKUP.WKEN，并选择唤醒事件 SYSCFG_WKUP.WKEG，可以是上升沿或下降沿模式。

3.2 UART模块

注意事项 1: 自动波特率

建议在启用自动波特率侦测功能(UART_MCON.ABREN)时，同时启用重复侦测自动波特率(UART_MCON.ABRREPT)功能，以提高侦测的准确性和稳定性。若发生侦测波特率超时(UART_RIF.ABTO)的情况，波特率开关(UART_MCON.ABREN)会自动清除，此时使用者需重新启用自动波特率侦测功能。另外，如果自动波特率侦测成功但接收到的数据与预期不符，也需要重新启用自动波特率侦测功能(UART_MCON.ABREN)。

注意事项 2: 发送空中断处理方式

UART 的 TX 数据传输缓存包括 UART_TXDATA 寄存器和 TX 位移寄存器。预设情况下，UART_TXDATA 为空，当空中断发生时，如果数据写入 UART_TXDATA，数据会立即转移到 TX 位移寄存器。此时 UART_TXDATA 仍然是空的，再次发生空中断。默认空中断和第一个数据的空中断之间的时间很短，因此在中断处理流程中，需优先清除中断再填写 UART_TXDATA，以便更好地控制数据传输。

3.3 I2C模块

注意事项 1: 当总线 SDA 遇到异常下拉的波形时，可能出现传输状态错误

发生条件为，当 I2C 配置为主机，I2C_CON2.START 写 1 时，从机输出 SDA 低电平，I2C 可能会进入从机接收模式；当从机输出不受控的 SDA 低电平时，需在系统流程中加入软件超时机制，在 I2C_CON2.START 写后开始计时，若 I2C 通讯超时，则复位 I2C。 复位 I2C 的方式为 I2C_CON1.PE 写 0，重新配置 I2C，然后 I2C_CON1.PE 写 1。

注意事项 2: 最大传输 NBYTE

当重载模式关闭时，I2C 支持最大 2^{16}- 1 =65535 Bytes 的传输数量，分别配置 I2C_CON1.NBYTES 与 I2C_CON2.NBYTES 寄存器。

当重载模式开启时，最大传输 NBYTE 为：

- 传送模式下，NBYTE 最大设定值为 65535，当传输数量需求大于 65535 时，在传输前配置 I2C_CON2.RELOAD 位，当第一笔 65535 Bytes 传输完后，SCL 将被拉低，此时可重新填写第二笔传输的 NBYTES 数量；当不使用 NBYTES 重载模式时，需将 I2C_CON2.RELOAD 清除，如果开启 I2C_CON2.AUTOEND 功能，I2C 会在 I2C_CON2.RELOAD 清除后，自动发送 STOP 信号。
- 接收模式下，NBYTE 最大设定值为 255，当传输数量需求大于 255 时，在传输前配置 I2C_CON2.RELOAD 位，当第一笔 255 Bytes 传输完后，SCL 将被拉低，此时可重新填写第二笔传输的 NBYTES 数量；当不使用 NBYTES 重载模式时，需将 I2C_CON2.RELOAD 清除，如果开启 I2C_CON2.AUTOEND 功能，I2C 会在 I2C_CON2.RELOAD 清除后，自动发送 STOP 信号。

注意事项 3: I2C 控制寄存器 2 (I2Cx_CON2)访问限制

I2C 控制寄存器 2 (I2Cx_CON2)，必须按字(32 位)访问。

注意事项 4: 发送空中断处理方式

I2C TX 数据传输缓存，包含 I2C_TXDATA 寄存器与 TX 位移寄存器；预设 I2C_TXDATA 为空，空中断发生，当写数据进入 I2C_TXDATA 后，数据立即被转移到 TX 位移寄存器，此时 I2C_TXDATA 为空，再次发生空中断。默认空中断与第一个数据的空中断之间的时间很短，因此中断处理流程中，需优先清除中断再填写 I2C_TXDATA。

注意事项 5: I2C 外设作从机时，只支持一个从机对一个主机。不支持 I2C 总线上有多从机应用。

3.4 SPI模块

注意事项 1: 发送 FIFO 缓存中断处理方式

SPI 发送 FIFO 中断标志位分为两种，发送 FIFO 缓存空中断(SPI_IFM.TXE)与发送 FIFO 缓存低于阀值中断(SPI_IFM.TXTH)。当进入中断处理流程时，务必优先清除中断再填写 SPI_DATA。根据不同的应用，为了避免填写 SPI_DATA 时，发生 FIFO 溢出的错误操作，提供以下两种方式检查：

- 方法 1：确认 FIFO 剩余空间，读取 SPI_STAT.TXFLV 寄存器，表示 FIFO 已填入数据数量 (字节)，可判断再填入多少数据。
- 方法 2：确认 FIFO 是否已满，读取 SPI_STAT.TXF 寄存器，表示 FIFO 已满，不可再填入数据。

注意事项 2: 接收 FIFO 缓存中断处理方式

SPI 接收 FIFO 中断标志位分为三种，接收 FIFO 缓存非空中断(SPI_IFM.RXNE)、接收 FIFO 缓存满中断(SPI_IFM.RXF)与接收 FIFO 缓存超过阀值中断(SPI_IFM.RXTH)。FIFO 功能主要为了提升 SPI 传输效率，因此大多使用 RXF 与 RXTH 中断事件，效率比较显著；当进入中断处理流程时，务必优先读取(搬移) SPI_DATA，避免 FIFO 满溢造成数据丢失，提供以下两种方式检查：

- 方法 1：确认 FIFO 缓存空间，读取 SPI_STAT.RXFLV 寄存器，表示 FIFO 接收字节数量，可判断读取多少数据。
- 方法 2：确认 FIFO 是否已空，读取 SPI_DATA 数据，判断 SPI_STAT.RXNE 寄存器为 0，表示 FIFO 已空，数据已读取完毕。

注意事项 3: 从机模式下，发送数据操作方法

SPI 配置为从机模式时，填写 SPI_DATA 数据的依据，建议使用发送 FIFO 缓存低于阀值状态标志位(SPI_STAT.TXTH)或中断(SPI_IFM.TXTH)。

3.5 模拟电源

模拟稳压电源 VDDA，主要提供模拟/数字转换器(ADC)、比较器(CMP)、运算放大器(OPA) 、温度传感器(TSENSE) 与电阻分压电路(VRES) 使用。

注意事项 1: 电压开关流程

用户需将 SYSCFG_APWR.VDDA_PWREN 设定为 1，并等待约 10us 的稳定时间以确保稳定性。

3.6 CMP 模块

注意事项 1: 比较器(CMP) 参考电压量测

当 CMP 需要量测参考电压时，需配置 SYSCFG_APWR.VREFINT_EN 开启电源。开启参考电压电源后，需等待 10us 的稳定时间。

3.7 ADC 模块

注意事项 1: ADC 时钟(ADCCLK)配置

通过 RCU_CFG2.ADC_ KSRC 设定输入时钟的来源为系统频率、AHB 时钟或 PLL 时钟，并通过 ADC_CCR.PRESCALE 设定输入时钟的分频比例，以生成 ADC 时钟。若需要更改时钟来源或分频配置，必须先将 RCU_CFG2.ADC_ KSRC 设置为 0，再进行修改，以确保输入频率的准确性。

注意事项 2: ADC 参考电压电源开关

当 ADC 需要量测参考电压时，需配置 SYSCFG_APWR.VREFINT_EN 开启电源。开启参考电

压电源后，需等待 10us 的稳定时间。ADC 的共模电压(VCM)为输入信号的参考电压。

注意事项 3:

VRES1 与 VRES2 分别连接到 ADC1 与 ADC2 通道 18，由于通道本身会有 1/4 VDD 的电压，会导致 VRES1 与 VRES2 电压被抬升，在使用时需注意。

3. 8 温度传感器模块

注意事项 1: 温度传感器配置

温度传感器开关通过 SYSCFG_APWR.VTSENSE_EN 开启。

3. 9 OPAMP模块

注意事项 1: 使用内部增益模式

通过配置 OPAINTOEN 为 1，将 OPA 输出信号送至 ADC 通道 12~15 进行采样处理。

注意事项 2: 外部增益模式

通过配置 OPAINTOEN 为 0，将 OPA 输出至 IO 使用；若此时须使用 ADC 进行采样，则可通过配置 IOMUX 使用 ADC 通道 0~3 进行采样。只有 OPA1、OPA2 支持外部增益模式。

3. 10 VRES 模块

使用电阻分压技术来依据模拟电源(VDDA)进行电压分压。分压后的信号分别连接至比较器(CMP)、运算放大器(OPAMP)及模拟数字转换器(ADC)。

注意事项 1:

VRES1 与 VRES2 分别连接到 ADC1 与 ADC2 通道 18，由于通道本身会有 1/4 VDD 的电压，会导致 VRES1 与 VRES2 电压被抬升，在使用时需注意。

第4章　系统电路与版图注意事项

4.1　最小系统电路

4.1.1　最小系统电路

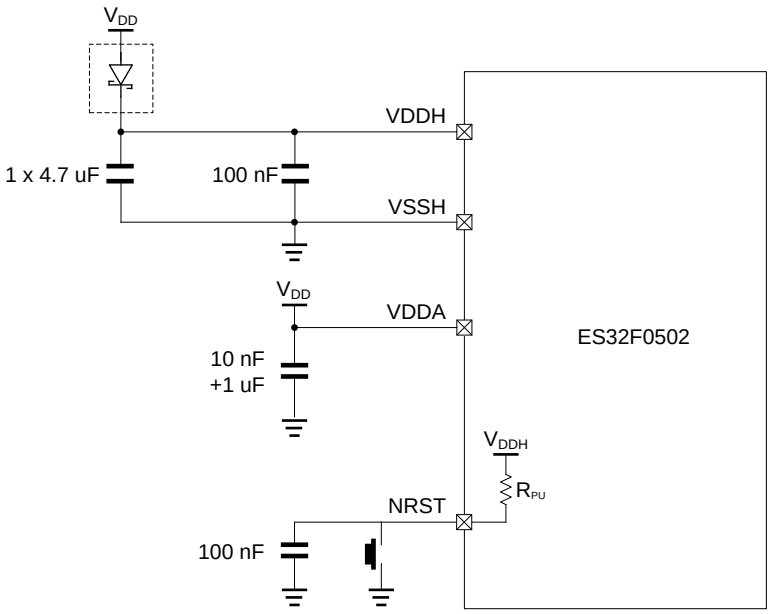

图 4-1　最小系统电路

注:

1. 芯片电源请务必依照图中建议进行配置，才能保证芯片电源在任何应用场景下都能够稳定。
2. 每一组电源必须连接陶瓷耦合电容(如图式)；这些电容必须尽可能靠近芯片的相应管脚，才能保证芯片的运行性能。
3. 需要为芯片提供稳定可靠的供电电源，当 VDD 电压波动时，电压跌落速率需慢于 200us/V，或者电压波动范围需小于±0.2V。如果 VDD 电源域有其他负载开关导致电压波动超出范围，可在 VDD 连接到 4.7μF 电解电容之前串联一个肖特基二极管。
4. NRST 引脚采用 RC 复位，支持内部上拉(上拉电阻值，请参考数据手册电器性章节)，电容采用 100nF。

4.2 外围电路

4.2.1 外部晶振HOSC与LOSC

注意事项 1: 外部陶瓷电容

电容建议数值，请参阅 ES32M0502 数据手册，电器特性章节。

注意事项 2: 版图参考原则

外部晶振应尽量靠近芯片，提高稳定度。

4.2.2 ESLINK II调试接口

注意事项 1: 电路建议

PB0 与 PB1 引脚预设为 ESLINK II 调试接口，PB1 为 SWCLK 默认为输入下拉模式，PB0 为 SWDIO 默认为输入上拉模式。当系统应用时，将此接口配置为其他 IO 功能时，如需维持 ESLINK II 调试接口的功能，务必预留 NRST 引脚功能，才能在 NRST=0 时，恢复 SWCLK 与 SWDIO 功能。

注意事项 2: 使用 Socket 进行 Flash 程序烧录时，芯片电源方案务必符合图 4-1 与图 4-2 的电源方案要求。

4.2.3 UART-BOOT烧录接口

注意事项 1: 开机程序默认选择

系统开机流程中，提供 BootRom 与 Main Flash 两种选择，而 ES32M0502 预设为 BootRom；此配置可通过修改配置字更换默认值。

注意事项 2: UART-BOOT 接口选用

当配置字 BOOT BYPASS 选项为 BootRom，表示系统开机从 BootRom 开始，PC12 与 PC13 引脚预设为 UART1 通信接口，PC12 为 UART1_TX 输出，PC13 为 UART1_RX 输入；由 UART 通信接口，搭配 UART-BOOT 烧录工具下载或更新程序。

当配置字 BOOT BYPASS 选项为 Main Flash，表示系统开机直接转跳至 Main 区执行，PC12 与 PC13 引脚预设为 GPIO，不具备 UART-BOOT 烧录功能。