文档编号：AN2055

上海东软载波微电子有限公司

# 应用笔记

## ES32F362x

# 修订历史

| 版本 | 修订日期 | 修改概要 |
|---|---|---|
| V1.0 | 2024-09-13 | 初版 |
| | | |
| | | |
| | | |

地　址：上海市徐汇区古美路 1515 号凤凰园 12 号楼 3 楼
邮　编：200235
E-mail：support@essemi.com
电　话：+86-21-60910333
传　真：+86-21-60914991
网　址：http://www.essemi.com/

# 目 录

## 内容目录

# 第1章 概述

## 1. 1 开发环境

推荐用户使用 Keil5 、IAR8.11 或者 GCC 进行固件开发。由于 Keil4 不支持 PACK 机制，故不推荐用户使用 Keil4。使用 ESLINK 或 Jlink 第一次烧录，需进行全擦操作。

## 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，适合大部分用户；

MD：基本上只提供寄存器位域级别的“读”、“写”接口，适合对芯片底层较为熟悉的用户。

如果用户对速度不是要求非常严格，一般情况下推荐用户使用 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

为避免程序的异常导致运行错误，芯片写保护寄存器用于阻止对被保护的寄存器误操作。

系统控制单元，GPIO，RTC，WDT 等模块支持寄存器写保护，对被保护的寄存器进行写之前需要解除写保护状态（允许写），否则无法对写保护寄存器写入。操作完成后，再使能写保护（禁止写）。库函数中均提供相应宏定义进行解除保护和使能保护。

### 1. 3. 1 系统写保护

系统控制寄存器的访问操作会影响整个芯片的运行状态，芯片提供系统设置保护寄存器 SYSCFG_PROT。对 SYSCFG_PROT 寄存器以字方式写入 0x55AA6996 会解除写保护， 对该寄存器写入其他任何值都会使能写保护。

可通过读 SYSCFG_PROT 寄存器确认写保护状态，读出值为 0x1，表示当前处于写保护状态；读出值为 0x0 表示当前处于写保护解除状态。

SYSCFG_PROT 保护的寄存器为除 SYSCFG_PROT 寄存器外的 SYSCFG、PMU、CMU、RMU 模块所有寄存器。

### 1. 3. 2 RTC 写保护

对 RTC_WPR 寄存器以字方式写入 0x55AAAA55 会解除写保护，写入其他值使能写保护。

可通过读 RTC_WPR 寄存器确认 RTC 模块是否处于写保护状态，读出值为 0x1，表示当前处于写保护状态；读出值为 0x0 表示 RTC 模块处于写保护解除状态。

该寄存器保护除自身外的 RTC 所有寄存器。

### 1. 3. 3 IWDT 写保护

对 IWDT_LOCK 寄存器以字方式写入 0x1ACCE551 会解除写保护，写入其他值使能写保护。

可通过读 IWDT_LOCK 寄存器确认 IWDT 模块是否处于写保护状态，读出值为 0x1，表示当前处于写保护状态；读出值为 0x0 表示 IWDT 模块处于写保护解除状态。该寄存器保护除自身外的 IWDT 所有寄存器。

### 1. 3. 4 WWDT 写保护

对 WWDT_LOCK 寄存器以字方式写入 0x1ACCE551 会解除写保护，写入其他值使能写保护。

可通过读 WWDT_LOCK 寄存器确认 WWDT 模块是否处于写保护状态，读出值为 0x1，表示当前处于写保护状态；读出值为 0x0 表示 WWDT 模块处于写保护解除状态。

该寄存器保护除自身外的 WWDT 所有寄存器。

## 1.4 写 1 清零寄存器

中断标志寄存器都是用“写 1 清零”的方式来操作。对于“写 1 清零”的寄存器，不可使用“读-修改-写”的方式来进行“写 1 清零”，否则会引起标志误清，进而产生漏中断的后果。对该类寄存器操作需要以字方式进行写。

## 1.5 位带操作

位带扩展区把每个 bit 扩展为一个 32-bits 的字，通过访问这些字可达到访问原始 bit 的目的；某个 bit 所在字的地址为 A，位序号为 N(0≤N≤31)，则该 bit 位带扩展后的地址为：

SRAM：AliasAddr=0x22000000+(A-0x20000000)x32+Nx4

外设：AliasAddr=0x42000000+(A-0x40000000)x32+Nx4

库函数中提供位带操作 API：

RAM 位带：void BITBAND_SRAM(uint32_t *addr, uint32_t bit, uint32_t val);

外设位带：void BITBAND_PER(volatile uint32_t *addr, uint32_t bit, uint32_t val);

# 第2章 系统控制

## 2.1 系统时钟

系统上电默认使用内部 4MHz 高速时钟(HRC)作为系统时钟。

可以使用 API 获取系统时钟：system_clock = ald_cmu_get_sys_clock();

### 2.1.1 内部高速 4MHz(默认时钟)

此种系统时钟不需要用户做任何配置。

### 2.1.2 外部时钟 HOSC(4~16MHz)

外部高速时钟要求为 4MHz 的倍数，如：4MHz、8MHz、12MHz、16MHz。

首先要确认焊接了外部高速时钟，并已知外部高速时钟的频率，假如外部高速时钟为 12MHz，则配置方式如下：

ald_cmu_clock_config(CMU_CLOCK_HOSC, 12000000);

使用外部时钟时，芯片进行软件 CHIP 复位或看门狗复位重启时，程序正常运行前需要增加一定的延时，确保外部晶振能正常起振。

### 2.1.3 72MHz

使用 HRC 倍频，配置方式如下：

ald_cmu_pll_config(ALD_CMU_PLL_INPUT_HRC, ALD_CMU_PLL_OUTPUT_72M);

ald_cmu_clock_config(ALD_CMU_CLOCK_PLL, 72000000);

使用 HOSC 倍频，外部高速时钟要求为 4MHz 的倍数，如：4MHz、8MHz、12MHz、16MHz。

首先要确认焊接了外部高速时钟，并已知外部高速时钟的频率，假如外部高速时钟为 12MHz，则配置方式如下：

ald_cmu_pll_config(ALD_CMU_PLL_INPUT_HOSC_3, ALD_CMU_PLL_OUTPUT_72M);

ald_cmu_clock_config(ALD_CMU_CLOCK_PLL, 72000000);

**注意事项 1：**系统时钟配置为 72Mhz 时，MSC_MEMWAIT.SRAM_W 至少需要设置为 1。

**注意事项 2：**系统时钟配置为 72Mhz 时，FLASH 的等待时间至少需要设置为 3，即 MSC.MEMWAIT.FLASH_W 在系统时钟为 72Mhz 时不能小于 3。对 MSC.MEMWAIT.FLASH_W 的设置需要放在 main 函数的第一行。

**注意事项 3：**SDK 系统时钟初始化后，各总线默认分频时钟：

| 系统时钟 | HCLK1 | HCLK2 | PCLK1 | PCLK2 |
|---|---|---|---|---|
| 72Mhz | - | - | - | 1/2 |
| 64Mhz | - | - | - | 1/2 |
| 56Mhz | - | - | - | 1/2 |
| 48Mhz | - | - | - | 1/2 |

表 2-1 系统各总线默认分频

注 1：使用 ALD 库函数时，库函数已对上述各总线时钟进行相应分频；

注 2：若没有特殊需求，建议外部高速晶振使用 12MHz。

### 2.1.4 外部低速时钟(LOSC)

首先要确认焊接了外部低速时钟及匹配电容，配置方式如下：

ald_cmu_clock_config(ALD_CMU_CLOCK_LOSC, 32768);

当系统时钟配置为低速时钟时(低于 1MHz)，SysTick 中断将会被迫关闭，ALD 提供的延迟类函

数禁止使用。

## 2.2 外部晶振

外部高速振荡器的典型应用连接：

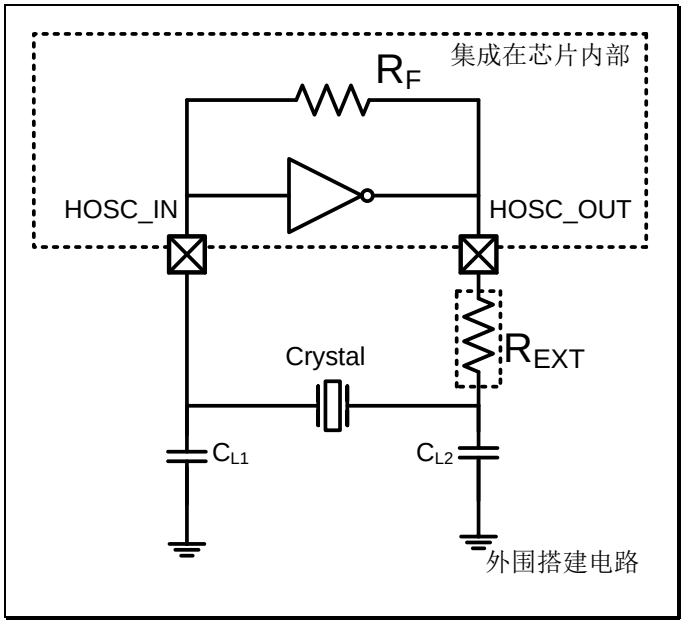


图 2-1 外部高速振荡器连接图

注 1：$R_{EXT}$ 阻值取决于晶振规格特性；

## 2.3 IAP 操作程序

芯片内置 IAP 自编程固化模块，由硬件电路实现。推荐使用 IAP 方式对 FLASH 进行擦、写操作，可以减少用户代码量。

## 2.4 FLASH 模块

**注意事项 1：**读保护规则描述：当 FLASH 的读保护级别设置为 level1 或 level2 时，运行在 SRAM 上的程序不能有读 FLASH 操作。

**典型应用 A：**程序运行在 SRAM 上，响应中断请求。将中断向量表拷贝至 SRAM 中，并设置中断向量偏移地址（内核寄存器 SCB->VTOR，地址为 0xE000ED08）。

**注意事项 2：**MSC_MEMWAIT 寄存器的 FLASH_W 位配置访问 Flash 时的等待系统时钟数。常用系统时钟访问 Flash 时的等待时钟数如下表所示：

| 系统时钟 | 72MHz | 64 MHz | 56 MHz | 48MHz | 4Mhz |
|---|---|---|---|---|---|
| FLASH_W | ≥3 | ≥2 | ≥2 | ≥1 | 无需等待 |

表 2-2 常用系统时钟访问 Flash 等待时钟数

**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 编程前必须先进行擦除。

## 2.5 配置字

**注意事项 1：**使用 ESBurner 擦除芯片后，需要执行配置字编程，否则将导致芯片运行不正常。即点击“擦除”按钮后，需要点击“配编”按钮。

**注意事项 2：**系统配置字默认使能硬件独立看门狗。如下图所示：

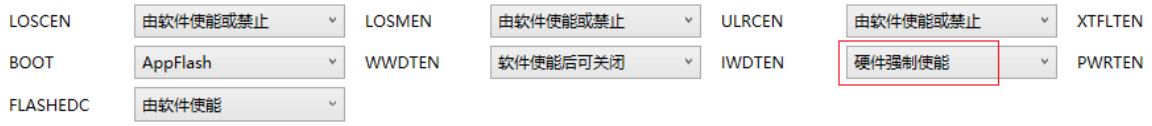


图 2-2 配置字默认使能 IWDT

# 第3章 外设

## 3. 1 GPIO 模块

**注意事项 1：**未使用的 IO 管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：**使能 GPIO 的 EXTI 中断时，外部 10ns 以内的脉冲可能会触发中断，且中断标志无法清除。规避方式参考 ald_gpio_exti_clear_flag_status();接口函数。

## 3. 2 CAN 模块

**注意事项 1：**当 MCU 使用中断方式接收，通信速率是 1Mbits/s，并且帧间隔小于 30uS 时，则 MCU 的主频应不小于 48MHz，否则会有丢帧风险。

**注意事项 2：**通信速率范围 10Kbits/s--1Mbits/s。当速率低于 20Kbits/s 时，要求 PCLK1 频率不超过 24MHz。

**注意事项 3：**由于 CAN 标准中要求时钟精度在 1.58%以内，故当产品的使用环境比较恶劣时，建议使用外部高精度晶振。外部晶振推荐使用 12MHz(和 ES-PDS 板保持一致)。

## 3. 3 IIC 模块

**注意事项 1：**IIC 作从机时，只支持一个从机对一个主机。不支持 IIC 总线上有多从机应用。

**注意事项 2：**IIC 的 STAT 寄存器读后，RXUD[8]、RXOV[7]、TXUD[3]、TXOV[2]位会自动复位。

## 3. 4 UART 模块

**注意事项 1：**读 STAT 寄存器后，TFOERR[18]、RFUERR[13]、RFOERR[12]位会自动复位。

**注意事项 2：**RIF 寄存器中与 FIFO 相关标志位的置位及复位条件：

| 标志[位] | 置位条件 | 复位条件 |
|---|---|---|
| TFEMPTY[16] | 发送 FIFO 空 | 发送 FIFO 非空 |
| TFTH[15] | 发送 FIFO 小于或等于阈值 | 发送 FIFO 大于阈值 |
| RFFULL[11] | 接收 FIFO 满 | 接收 FIFO 不满 |
| RFNEMPTY[10] | 接收 FIFO 非空 | 接收 FIFO 空 |
| RFTH[9] | 接收 FIFO 大于等于阈值 | 接收 FIFO 小于阈值 |

表 3-1 UART FIFO 标志位置位和复位

## 3. 5 TIMER 模块

**注意事项 1：**使用外部时钟源（ET 引脚）时，中心计数模式时，计数无法触发更新事件。

## 3. 6 SPI 模块

**注意事项 1：**SPI 作主机，使用 DMA 发送时，不支持 CRC 校验。

**注意事项 2：**仅 SPI0 支持四线模式。使用四线模式时，SPI0 的最大时钟为 $f_{PCLK1}/4$。

**注意事项 3：**SPI 的 STAT 寄存器读后，RXUD[11]、RXOV[10]、TXUD[3]、TXOV[2]位会自动复位。

**注意事项 4：**系统时钟 72Mhz 时，SPI 作主/从机，I/O 配置为强驱，发送和接收的最大速率（特征值）如下表所示：

| 外设 | 模式 | | 作主机 | | 作从机 | |
|---|---|---|---|---|---|---|
| | | | 发送 | 接收 | 发送 | 接收 |
| SPI | CPOL = 1 | CPHA = 1 | $f_{PCLK1}$/2 | $f_{PCLK1}$/2 | $f_{PCLK1}$/4 | $f_{PCLK1}$/4 |
| | 其它模式 | | $f_{PCLK1}$/4 | $f_{PCLK1}$/4 | $f_{PCLK1}$/4 | $f_{PCLK1}$/4 |

表 3-2 SPI 作主/从机时的最快速率

**注意事项 5：**RIF 寄存器中与 FIFO 相关标志位的置位及复位条件：

| 标志[位] | 置位条件 | 复位条件 |
|---|---|---|
| RXTHRI[12] | 接收 FIFO 超过阈值 | FIFO 数据低于阈值 |
| RXFRI[9] | 接收 FIFO 满 | FIFO 不满 |
| RXNE[8] | 接收 FIFO 非空 | 接收 FIFO 空 |
| TXTHRI[4] | 发送 FIFO 低于阈值 | 发送 FIFO 大于或等于阈值 |
| TXERI[0] | 发送 FIFO 空 | 发送 FIFO 非空后 |

表 3-3 SPI FIFO 标志位置位和复位

**注意事项 6：**系统时钟 72Mhz，SPI 通信时钟为 $f_{PCLK1}$/2，且使用 DMA 时，需将对应通道 DMA_CHxCON 寄存器的 single 位复位。可参考 SDK 中 DMA 的 04_perh_to_perh_dma_flash 例程。

**注意事项 7：**系统时钟 72Mhz，SPI 配置为单工模式，且使用 DMA 时，需将对应通道 DMA_CHxCON 寄存器的 single 位复位，SPI 通信时钟不能高于 $f_{PCLK1}$/4。可参考 SDK 中 DMA 的 05_perh_to_perh_dma_spi_rxo 例程。

## 3. 7 ADC 模块

**注意事项 1：**系统运行后，调用 ald_get_adc_gain_offset()函数可获取 ADC 通道 1 到通道 16 的 gain 和 offset 补偿值。ADC 转换完成后，将转换值、gain 及 offset 值带入 ald_adc_value_adjust()函数进行高速或低速软件补偿。

对 ADC 转换值软件补偿是牺牲效率换精度的过程。如表 3-4 所示，常温下，$V_{REF}$=VDD=5V，$f_{ADC}$ = 36Mhz，$f_s$ = 2Mhz 时，不补偿、高速补偿及低速补偿的对比：

| | 不补偿-ADC 原始值 | | | 高速补偿 | | | 低速补偿 | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| 参数项 | DNL | INL | 精度误差 | DNL | INL | 精度误差 | DNL | INL | 精度误差 | UNIT |
| 典型值 | 2 | 7 | 14 | 2 | 6 | 10 | 2 | 5 | 7 | LSB |
| 转换速率 | 2Mhz | | | 400Khz | | | 200Khz | | | – |

表 3-4 ADC 值补偿对比

通过 ald_adc_normal_get_value()/ald_adc_insert_get_value()函数获取 normal/insert 通道不补偿的原始值，ald_adc_value_adjust(gain,offset,buffer,len, ADC_TRIM_MODE_FAST)/ ald_adc_value_adjust(gain,offset,buffer,len, ADC_TRIM_MODE_SLOW)函数可进行高速/低速补偿，是否选择及选择哪种补偿方案，可根据具体应用选择。

**注意事项 2：**为了保证 ADC 转换结果的稳定可靠、避免噪声干扰，建议在模拟输入通道对地挂接

100nF 或 10nF 电容进行滤波。

## 3. 8 内部温感

**注意事项 1：**ADC 通道 17 可测量芯片温度。调用 ald_get_temp_k()函数获取温度转换系数 k。将通道 17 的 ADC 转换值和系数 k 带入 ald_adc_tsensor_get_temperature()函数，返回值为芯片温度

# 第4章　最小系统电路

## 4. 1　ES32F362x LQFP64 封装芯片最小系统电路

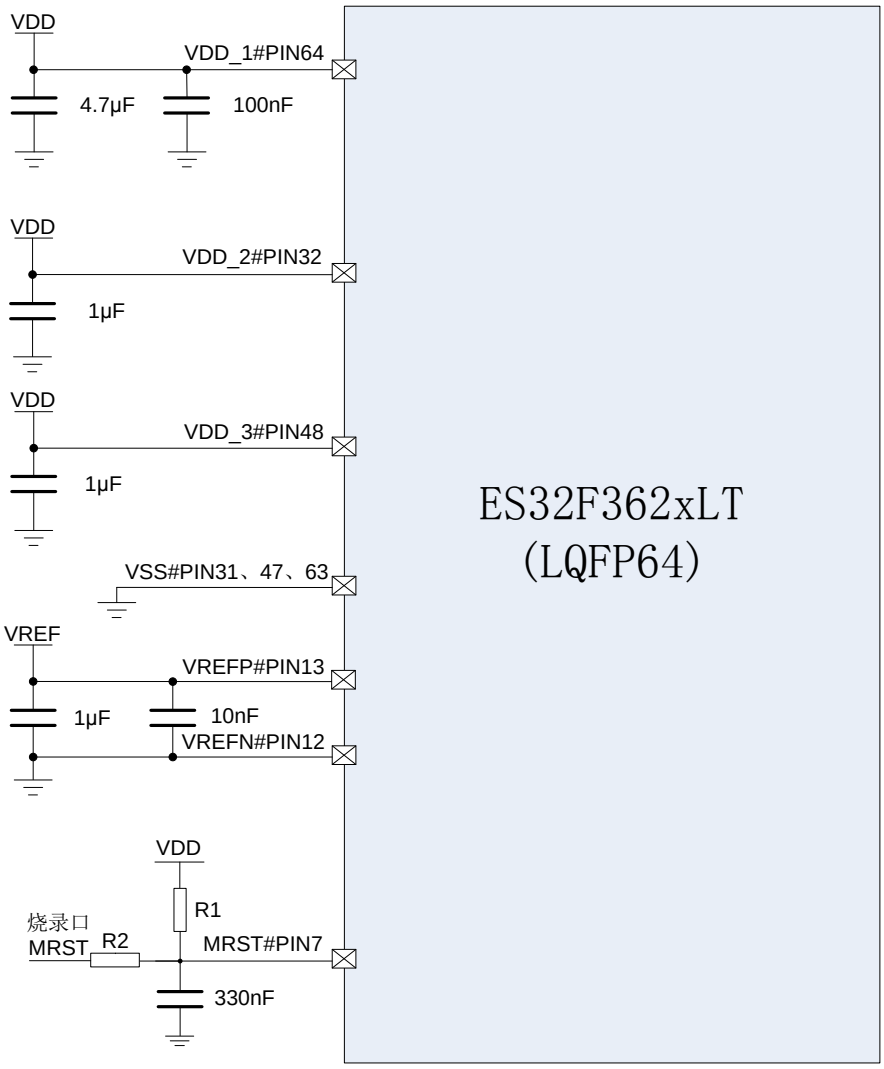


图 4-2 ES32F362x LQFP64 封装芯片最小系统电路

注 1：每一组电源必须连接如图所示的陶瓷耦合电容。这些电容必须尽可能地靠近芯片的相应管脚，才能保证芯片的运行性能。

注 2：MRST 引脚采用 RC 复位，其中 47KΩ≤R1≤100KΩ，电容 330nF≤C1≤1uF，R2 为限流电阻，0.1KΩ≤R2≤1KΩ

## 4. 2 ES32F362x LQFP48 封装芯片最小系统电路

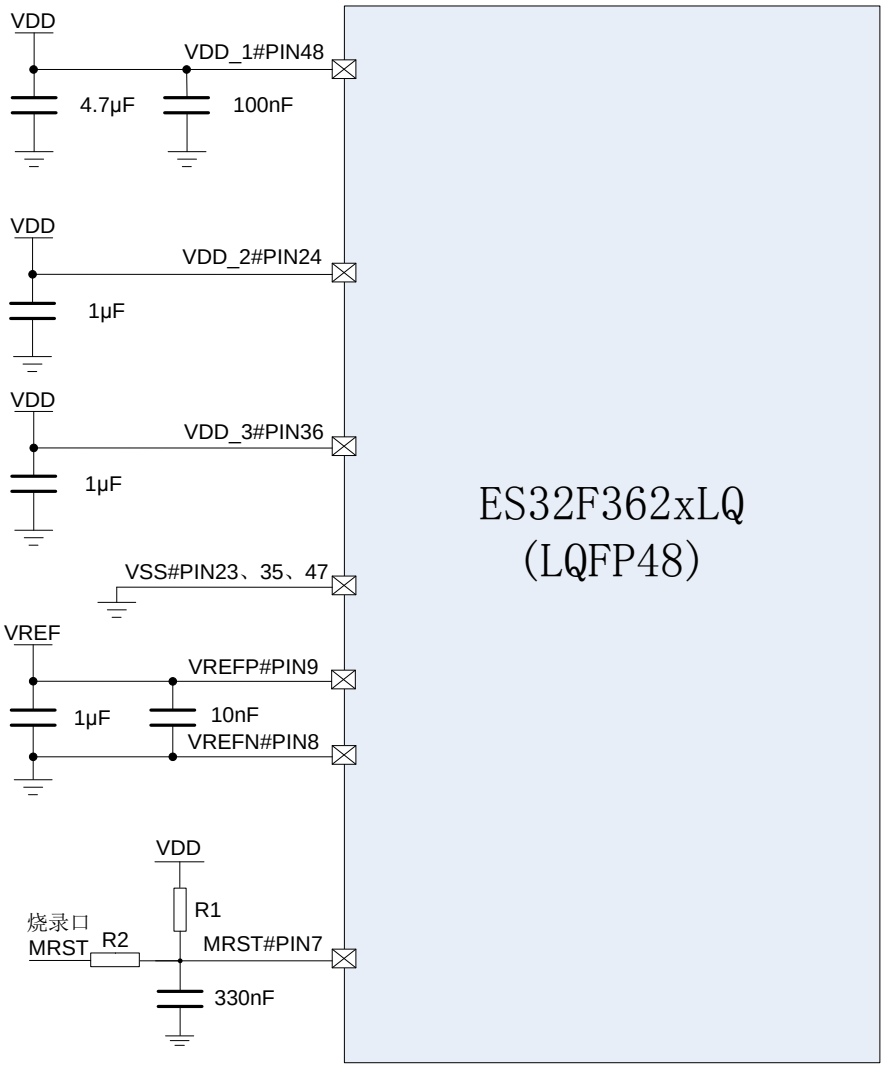


图 4-4 ES32F362x LQFP48 封装芯片最小系统电路

注 1：每一组电源必须连接如图所示的陶瓷耦合电容。这些电容必须尽可能地靠近芯片的相应管脚，才能保证芯片的运行性能。

注 2：MRST 引脚采用 RC 复位，其中 47KΩ≤R1≤100KΩ，330nF≤C1≤1uF，R2 为限流电阻，0.1KΩ≤R2≤1KΩ。