

GigaDevice Semiconductor Inc.

GD32H75E 芯片使用限制

勘误手册

1.0 版本

(2026 年 10 月)

目录

目录.....	2
图索引.....	4
表索引.....	5
1. 前言.....	6
1.1. 芯片版本定义.....	6
1.2. 芯片使用限制总览.....	6
2. 芯片使用限制描述.....	8
2.1. SYSTEM.....	8
2.1.1. 系统运行在高主频且中断向量表和中断处理代码均放置在 ITCM 中时, 高频中断会概率性导致 ITCM ECC 误报.....	8
2.2. GPIO.....	8
2.2.1. 当 VDD 和 VDDA 掉电后, 如果继续向 GPIO(除 PA9 / PA10 / PB12 / PB13 外)注入电压, 可能导致芯片 GPIO 出现过流风险.....	8
2.3. TIMER.....	9
2.3.1. TIMER 中断存在误触发的风险.....	9
2.4. USART.....	9
2.4.1. LIN 模式下会发生数据采样错误.....	9
2.4.2. 在静默模式下, 非唤醒帧引起的奇偶校验错会导致 PERR 置位.....	9
2.5. I2S.....	10
2.5.1. 当使用 I2S1 进行通信且 GPIO 和 I2S1 配置处于特殊组合时, 可能会导致 I2S1_CK 和 I2S1_WS 引脚持续输出信号.....	10
2.6. OSPI.....	10
2.6.1. 当 OSPI 仅发送数据段时, 第一个时钟会丢失.....	10
2.6.2. 当 OSPI 运行时钟大于 100MHz 时, 在状态轮询模式下读取外部存储器状态标志异常.....	10
2.7. EXMC.....	11
2.7.1. 在访问 SDRAM 期间, 可能会出现总线卡住的情况.....	11
2.7.2. 不支持非对齐地址访问.....	11

2.8. CAN	11
2.8.1. 在接收邮箱处理过程中, CAN RAM 区域可能会被篡改	11
2.8.2. 在总线离线恢复后, 发送错误计数未自动清零	12
2.8.3. 邮箱数据未及时读走, 可能会导致当前帧和下一帧数据读取错误	12
2.8.4. 当 CAN 运行时钟频率低于 CK_APB2 且 DLC 段发生错误时, 会导致接收到错误的帧 ID	13
2.9. ESC	13
2.9.1. 默认配置下, 外部 MCU(I2C 主机)无法通过 ESC 芯片的 EESDA / EESCL 引脚访问 EEPROM	13
2.9.2. IRQ 中断函数不响应, 中断无法触发	14
2.9.3. EtherCAT PHY 同步上电 link 失败	14
2.10. Core	18
3. 版本历史	19

图索引

图 1-1. GD32H75E 的芯片修订版本.....	6
------------------------------	---

表索引

表 1-1. 适用产品	6
表 1-2. 芯片使用限制	6
表 3-1. 版本历史	19

1. 前言

该文档适用于 GD32H75E 产品，具体参考[表 1-1. 适用产品](#)。该文档提供了在使用 GD32 MCU 过程中需要注意的技术细节，以及相关问题的解决方案。

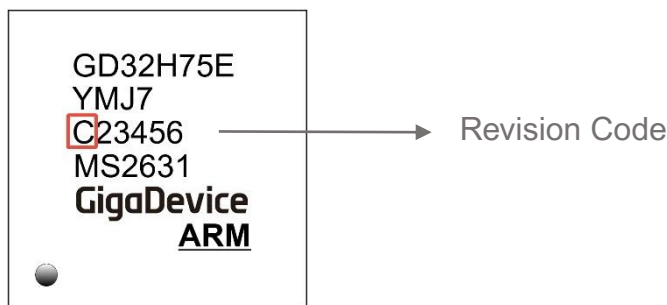
表 1-1. 适用产品

类型	产品系列
MCU	GD32H75Exx 系列

1.1. 芯片版本定义

可通过芯片丝印上的标记来确定当前芯片的版本。丝印上第 3 行的第 1 个编码表示芯片当前版本，如[图 1-1. GD32H75E 的芯片修订版本](#)所示。

图 1-1. GD32H75E 的芯片修订版本



1.2. 芯片使用限制总览

GD32H75E 芯片使用限制参考[表 1-2. 芯片使用限制](#)。关于芯片使用限制的更多详细信息请参考第 2 章。

表 1-2. 芯片使用限制

模块	使用限制	解决方案
		修订版本 C
SYSTEM	系统运行在高主频且中断向量表和中断处理代码均放置在 ITCM 中时，高频中断会概率性导致 ITCM ECC 误报	Y
GPIO	当 VDD 和 VDDA 掉电后，如果继续向 GPIO(除 PA9 / PA10 / PB12 / PB13 外)注入电压，可能导致芯片 GPIO 出现过流风险	Y
TIMER	TIMER 中断存在误触发的风险	Y
USART	LIN 模式下会发生数据采样错误	Y
	在静默模式下，非唤醒帧引起的奇偶校验错会导致 PERR 置位	Y
I2S	当使用 I2S1 进行通信且 GPIO 和 I2S1 配置处于特殊组合时，可能会导致 I2S1_CK 和 I2S1_WS 引脚持续输出信号	Y
OSPI	当 OSPI 仅发送数据段时，第一个时钟会丢失	Y

模块	使用限制	解决方案
		修订版本 C
	当 OSPI 运行时钟大于 100MHz 时, 在状态轮询模式下读取外部存储器状态标志异常	Y
EXMC	在访问 SDRAM 期间, 可能会出现总线卡住的情况	Y
	不支持非对齐地址访问	Y
CAN	在接收邮箱处理过程中, CAN RAM 区域可能会被篡改	Y
	在总线离线恢复后, 发送错误计数未自动清零	Y
	邮箱数据未及时读走, 可能会导致当前帧和下一帧数据读取错误	Y
	当 CAN 运行时钟频率低于 CK_APB2 且 DLC 段发生错误时, 会导致接收到错误的帧 ID	Y
ESC	默认配置下, 外部 MCU(I2C 主机)无法通过 ESC 芯片的 EESDA/ EESCL 引脚访问 EEPROM	Y
	IRQ 中断函数不响应, 中断无法触发	Y
	EtherCAT PHY 同步上电 link 失败	Y

注意:

Y = 存在使用限制且存在解决方案

N = 存在使用限制但不存在解决方案

'-' = 使用限制已修复

2. 芯片使用限制描述

2.1. SYSTEM

2.1.1. 系统运行在高主频且中断向量表和中断处理代码均放置在 ITCM 中时，高频中断会概率性导致 ITCM ECC 误报

描述与影响

当系统时钟配置为高频(如 600MHz)，AXI/AHB 时钟配置为系统时钟的二分频且 TCM 插入等待周期时(TCM_WAITSTATE = 1)，如果将中断向量表和中断处理代码都放置在 ITCM 中，若有高频中断产生，会概率性导致 ITCM ECC 误报。

注意：当系统时钟高于 350MHz 时，TCM 需要插入等待周期。

解决方案

使用以下解决方案之一：

- 1) 将中断向量表放置在 AXISRAM 中。
- 2) 避免将中断向量表和中断处理代码都放在同一块 TCM 中。例如，可以将中断向量表放置在 DTCM 中，中断处理代码放在 ITCM 中。
- 3) 采用降频方案。避免 TCM_WAITSTATE = 1 和 AXI/AHB 时钟配置为系统时钟的二分频同时满足。如系统时钟和 AXI/AHB 时钟均配置为 300MHz。

2.2. GPIO

2.2.1. 当 VDD 和 VDDA 掉电后，如果继续向 GPIO(除 PA9 / PA10 / PB12 / PB13 外)注入电压，可能导致芯片 GPIO 出现过流风险

描述与影响

当芯片的 VDD 和 VDDA 掉电后，如果继续向 GPIO(除 PA9 / PA10 / PB12 / PB13 外)外部注入电压(Vin)，由于 GPIO 的 ESD 保护电路，外部注入的电压可能会泄漏到 VDD，形成 Vin-0.7V 的电压。如果 Vin-0.7V 接近 POR(1.6V)，可能会满足芯片的 POR 条件。然而，由于 GPIO 的过流能力有限，这可能导致系统发生重复的 POR/PDR 事件，从而引发系统不可预测的风险。

解决方案

在系统设计中，应在芯片的 VDD 和 NRST 之间外接一个欠压复位芯片，以确保当 GPIO 泄漏到 VDD 的电压低于欠压复位芯片的阈值电压时，可以直接拉低 NRST 以避免错误操作。具体解决方案可参考《AN225 GD32H7xx 电源旁路模式使用指南》。

2.3. TIMER

2.3.1. TIMER 中断存在误触发的风险

描述与影响

当启用 TIMER 中断且 MCU 以高频率运行时，由于代码执行速度较快且清除中断标志需要一定的时间，可能导致无法及时清除相应的 TIMER 中断标志，从而存在重复进入 TIMER 中断的风险。

解决方案

在 TIMER 中断处理函数的开始位置清除相应的 TIMER 中断标志，并在退出中断前保持至少 20 个指令周期。

2.4. USART

2.4.1. LIN 模式下会发生数据采样错误

描述与影响

当 USART 作为接收器处于 LIN 模式时，由于在接收断开帧之前启用了自动波特率检测功能，会发生数据采样错误。

解决方案

- 1) 在接收断开帧之前禁用自动波特率检测功能。
- 2) 在置位 LIN 断开帧检测标志(LBDF)后启用自动波特率检测功能。

2.4.2. 在静默模式下，非唤醒帧引起的奇偶校验错会导致 PERR 置位

描述与影响

在静默模式下，由非唤醒帧引起的奇偶校验错误会导致奇偶校验错误(USART_STAT 寄存器中的 PERR 位被设置)，即使唤醒帧中没有发现奇偶校验错误。

解决方案

软件忽略在这种情况下生成的奇偶校验错误标志。

2.5. I2S

2.5.1. 当使用 I2S1 进行通信且 GPIO 和 I2S1 配置处于特殊组合时，可能会导致 I2S1_CK 和 I2S1_WS 引脚持续输出信号

描述与影响

当使用 I2S1 进行通信，且 GPIO 和 I2S1 配置为特殊组合时，I2S1_CK 和 I2S1_WS 引脚将概率性地连续输出信号。具体配置及现象如下：

GPIO 配置	I2S1 配置	现象
GPIO_PUPD_PULLDOWN	I2S_CKPL_HIGH	I2S1_CK 和 I2S1_WS 连续发波
GPIO_PUPD_NONE	I2S_CKPL_HIGH	I2S1_CK 和 I2S1_WS 连续发波
GPIO_PUPD_PULLUP	I2S_CKPL_HIGH	I2S1 工作正常
GPIO_PUPD_PULLDOWN	I2S_CKPL_LOW	I2S1 工作正常
GPIO_PUPD_PULLUP	I2S_CKPL_LOW	I2S1 工作正常
GPIO_PUPD_NONE	I2S_CKPL_LOW	I2S1 工作正常

解决方案

GPIO 和 I2S1 的正确配置请参考上述。

2.6. OSPI

2.6.1. 当 OSPI 仅发送数据段时，第一个时钟会丢失

描述与影响

当 OSPI 仅发送数据段时，第一个时钟会丢失，从而导致第一个数据丢失。

解决方案

将第一个数据作为命令发送。

2.6.2. 当 OSPI 运行时钟大于 100MHz 时，在状态轮询模式下读取外部存储器状态标志异常

描述与影响

当 OSPI 运行时钟大于 100MHz 时，由于硬件连续轮询间隔不足，外部存储器可能未准备好，随后 OSPI 将执行后续操作，从而导致异常。

解决方案

在每次轮询存储器状态之前添加硬件延迟(例如 20ms 延迟)。

2.7. EXMC

2.7.1. 在访问 SDRAM 期间，可能会出现总线卡住的情况

描述与影响

在访问 SDRAM 时，如果 SDRAM 数据位宽为 32 位，总线发送两个 8 位或两个 16 位访问，然后发送一个 64 位访问，如果 64 位访问的数据恰好由之前的两个 8 位或 16 位访问组成，则总线可能会卡住。

解决方案

使用以下解决方案之一：

- 1) 禁用 SDRAM 的突发访问(清除 EXMC_SDCTL0/1 寄存器的 BRSTRD 位)
- 2) 使能 CPU 缓存功能

2.7.2. 不支持非对齐地址访问

描述与影响

总线不支持对 SDRAM 的非对齐地址访问。

解决方案

在访问 SDRAM 之前启用 CPU 的缓存功能。

2.8. CAN

2.8.1. 在接收邮箱处理过程中，CAN RAM 区域可能会被篡改

描述与影响

如果在接收邮箱处理例程中未执行全局邮箱解锁操作(由于软件操作错误)，CAN RAM 区域有一定概率会被篡改，从而导致数据传输和接收异常。

解决方案

在每次数据传输之前，等待 CAN_STAT 寄存器中的发送完成标志置位，而不是通过发送邮箱的 CODE 段值进行判断。参考代码如下：

```
Flagstatus can_tx_state = RESET;
{
```

```

        if((RESET == can_tx_state) || (SET == can_flag_get(CAN1, CAN_FLAG_MB1))){
            can_tx_state = SET;
            can_flag_clear(CAN1, CAN_FLAG_MB1);
            /* transmit message */
            can_mailbox_config(CAN1, 1, &transmit_message);
            /* user code */
        }
    }
}

```

2.8.2. 在总线离线恢复后，发送错误计数未自动清零

描述与影响

在总线离线恢复后，发送错误计数(TECNT)未被硬件自动清零。

解决方案

在总线关闭恢复后，使用软件方法清除发送错误计数(TECNT)。以 CAN1 为例，参考代码如下：

```

{
    /* bus off recovery flag is set */
    if(RESET != can_flag_get(CAN1, CAN_FLAG_BUSOFF_RECOVERY))
    {
        /* enter inactive mode */
        can_operation_mode_enter(CAN1, CAN_INACTIVE_MODE);
        /* clear transmit error count */
        CAN_ERR0(CAN1) &= ~(CAN_ERR0_TECNT);
        /* enter normal mode */
        can_operation_mode_enter(CAN1, CAN_NORMAL_MODE);
        can_flag_clear(CAN1, CAN_FLAG_BUSOFF_RECOVERY);
    }
}

```

2.8.3. 邮箱数据未及时读走，可能会导致当前帧和下一帧数据读取错误

描述与影响

在处理邮箱接收时，如果在读取当前帧数据时，一个新的 CAN 帧(下一帧)被移入接收邮箱，可能导致当前帧和下一帧数据读取错误。

注意：仅影响当前帧和下一帧。

解决方案

使用以下解决方案之一：

- 1) 使用邮箱接收中断并将其配置为最高优先级。当接收邮箱中断发生时，及时读取并处理邮

箱数据(在下一帧结束之前)。

- 2) 使用 CAN FIFO 接收代替邮箱接收。
- 3) 通过置位 RPFQEN 使能邮箱队列，并将邮箱接收中断设置为最高优先级。

2.8.4. 当 CAN 运行时钟频率低于 CK_APB2 且 DLC 段发生错误时，会导致接收到错误的帧 ID

描述与影响

当 CAN 运行时钟频率低于 CK_APB2 时，如果 CAN 总线上的外部干扰导致在接收帧时，DLC 段检测到错误并发送错误帧，则会导致后续帧接收到错误帧 ID(即使该错误帧 ID 已被 CAN 滤波器过滤)。

解决方案

使用以下解决方案之一：

- 1) 软件将 CAN 时钟源配置为 CK_APB2。以 CAN1 为例，参考代码如下：

```
{  
    /* configure the CAN1 clock source as CK_APB2 */  
    rcu_can_clock_config(IDX_CAN1, RCU_CANSRC_APB2);  
}
```

- 2) 软件检查帧 ID(适用于 CAN 运行时钟频率低于 CK_APB2 的情况)。当 CAN 邮箱接收到非目标帧 ID 且发生错误时，软件需要重新配置 CAN 接收邮箱参数。具体配置步骤如下：
 - a) 进入未激活模式
 - b) 重新配置接收邮箱参数
 - c) 进入正常模式

2.9. ESC

2.9.1. 默认配置下，外部 MCU(I2C 主机)无法通过 ESC 芯片的 EESDA / EESCL 引脚访问 EEPROM

描述与影响

当使用外部 MCU 访问 EEPROM 时，由于 ESC 芯片的 EESDA / EESCL 引脚默认输出为推挽模式，会导致外部 MCU 无法读出 EEPROM 数据。以上情况下，ESC 芯片依然可正常访问 EEPROM，不影响 ESC 功能使用。

解决方案

外部 MCU 通过 PDI 接口配置 ESC 芯片的 EESDA / EESCL 引脚为输出开漏模式后，外部芯片才可以访问 EEPROM。

2.9.2. IRQ 中断函数不响应，中断无法触发

描述与影响

当 GD32H75E 系列 MCU 的 ESC 模块在 SM 模式运行时，若主站发帧周期抖动较大，相邻两帧数据间隔小于 ESC 中断处理耗时，会造成 IRQ 中断无法再次触发，从站无法再刷新主站下发的 PDO 数据；重新切换 OP 状态可恢复数据刷新。

解决方案

PJ10 为 ESC 硬件 IRQ 与 MCU 相连接的引脚。在 IRQ 中断服务函数内增加 PJ10 引脚电平检测：执行完 PDI_Isr() 并清除 ESC 中断标志后，若 PJ10 仍为低电平，代表仍存在未处理中断，再次调用 PDI_Isr() 处理滞留事件，并重新清除中断标志。

参考代码如下：

```
/* code example */
void EcatIsr(void)
{
    if(exti_interrupt_flag_get(EXTI_10) != RESET) {

        ACK_ESC_INT;
        PDI_Isr();

        if(gpio_input_bit_get(GPIOJ, GPIO_PIN_10) == RESET) {
            PDI_Isr();
            ACK_ESC_INT;
        }
    }
}
```

2.9.3. EtherCAT PHY 同步上电 link 失败

描述与影响

使用 GD32H75E 系列 MCU 内置 ESC 模块时，若总线上两个（或多个）节点上电时序完全一致，由于内置 PHY 的 FLP 协商脉冲基于伪随机时序生成，上电复位后这些节点的伪随机初始相位完全一致——前、后端 PHY 将同时发送 FLP，造成脉冲撞帧冲突，进而导致自协商无法立即 link-up。

两节点间只能依靠晶振频偏缓慢拉开相位差，待 FLP 发送时机错开后链路才能建立。现象上表现为部分从站 Link-Up 缓慢。

解决方案

上电软件启动后，生成一段 0–65535us 范围内的真随机延时；延时结束后再执行 PHY 软件复位，随后启动 EtherCAT 协议栈。

通过人为偏移各节点 PHY 的复位起点，打散 FLP 脉冲发送相位，避免同步撞帧，从而实现

多节点同时上电场景下 PHY 快速完成自协商、稳定建立链路。

参考代码如下：

```
/*!
 * \brief      enable esc func
 * \param[in]  none
 * \param[out] none
 * \retval     none
 */
static void esc_enable(void)
{
    volatile uint32_t temp = 0;

    ospi_hw_init(OSPI_INTERFACE, &ospi_init_struct);
    ospi_enable_ospi_mode(OSPI_INTERFACE, &ospi_init_struct, SPI_MODE);

    /*ByteTest and PMU ready*/
    ospi_read(PMU_PDIREFVAL, (uint8_t*)&temp, 4);
    if(temp != 0x76543210)
    {
        while(1);
    }

    trng_val= trng_get();
    delay_lms(20);          /* ensure the PHY hardware reset time */
    delay_us(trng_val);     /* 0-65535us random delay to stagger the PHY reset among slaves */

    temp = 0x307;
    ospi_write(ESC_RCU_RSTCFG, (uint8_t *)&temp, 4);
    delay_lms(5);

    ospi_enable_ospi_mode(OSPI_INTERFACE, &ospi_init_struct, SPI_MODE);

    ospi_read(PMU_CTRL0, (uint8_t*)&temp, 4);
    if( (temp & BIT(0)) != BIT(0))
    {
        while(1);
    }
    esc_oeext_clock_disable();
}
```

```

/*!
\brief      check whether the TRNG module is ready
\param[in]  none
\param[out] none
\retval    ErrStatus: SUCCESS or ERROR
*/
ErrStatus trng_ready_check(void)
{
    uint32_t timeout = 0;
    FlagStatus trng_flag = RESET;

    /* check whether the random data is valid */
    do {
        timeout++;
        trng_flag = trng_flag_get(TRNG_FLAG_DRDY);
    } while((RESET == trng_flag) && (0xFFFFU > timeout));

    if((SET == trng_flag)
        && (RESET == trng_interrupt_flag_get(TRNG_INT_FLAG_CEIF))
        && (RESET == trng_interrupt_flag_get(TRNG_INT_FLAG_SEIF))
        && (RESET == trng_flag_get(TRNG_FLAG_CECS))
        && (RESET == trng_flag_get(TRNG_FLAG_SECS))
        && (OU == (TRNG_STAT & TRNG_STAT_ERRSTA))) {
        return SUCCESS;
    }

    /* return check status */
    return ERROR;
}

```

```

/*!
 \brief      configure the TRNG module parameters
 \param[in]  none
 \param[out] none
 \retval     ErrStatus: SUCCESS or ERROR
 */
ErrStatus trng_configuration(void)
{
    uint32_t timeout = 0;

    /* configure the internal 48M high speed clock for TRNG */
    rcu_osc_on(RCU_IRC48M);
    /* wait until IRC48M clock is stable */
    while((RESET == rcu_flag_get(RCU_FLAG_IRC48MSTB)) && (0xFFFFU > timeout)) {
        timeout++;
    }
    if(RESET == rcu_flag_get(RCU_FLAG_IRC48MSTB)) {
        return ERROR;
    }

    /* select the internal IRC48M as the 48M system clock source */
    rcu_ck48m_clock_config(RCU_CK48MSRC_IRC48M);

    /* enable the TRNG peripheral clock */
    rcu_periph_clock_enable(RCU_TRNG);
    /* enable the TRNG clock error detection */
    trng_clockerror_detection_enable();

    /* reset the TRNG registers to default configuration */
    trng_deinit();
    /* enable the post-processing conditioning module reset */
    trng_conditioning_reset_enable();
    /* configure the TRNG working mode as NIST security mode */
    trng_mode_config(TRNG_MODSEL_NIST);
    /* configure the conditioning input bit width: 440bit raw entropy data */
    trng_conditioning_input_bitwidth(TRNG_INMOD_440BIT);
    /* configure the conditioning output bit width: 256bit, output eight 32-bit random numbers */
    trng_conditioning_output_bitwidth(TRNG_OUTMOD_256BIT);
    /* configure the conditioning crypto algorithm as SHA256 hash algorithm */
    trng_conditioning_algo_config(TRNG_ALGO_SHA256);
    /* enable the TRNG post-processing conditioning module */
    trng_conditioning_enable();
    /* enable the TRNG security post-processing */
    trng_postprocessing_enable();

    /* enable the TRNG hardware module */
    trng_enable();

    return SUCCESS;
}

```

```

/*!
\brief      get a 16-bit true random number from the TRNG module
\param[in]   none
\param[out]  none
\retval     low 16 bits of the random data; SysTick-derived value on failure
\note       The CK48M clock source and the IRC48M enable state are backed up
            before the TRNG is configured and restored before returning, so
            the system clock and other clock domains (e.g. USB) are left
            unchanged after this function returns.
*/
uint16_t trng_get(void)
{
    uint8_t retry = 0;
    uint16_t result = 0U;
    ErrStatus random_status = ERROR;
    /* backup the current CK48M clock source and IRC48M enable state */
    uint32_t ck48m_src_bak = RCU_ADDCTL0 & RCU_ADDCTL0_CK48MSEL;
    uint32_t irc48m_en_bak = RCU_ADDCTL0 & RCU_ADDCTL0_IRC48MEN;
    uint32_t trng_clk_en_bak = RCU_AHB2EN & RCU_AHB2EN_TRNGEN;

    /* retry random data acquisition up to three times */
    while((retry < 3U) && (ERROR == random_status)) {
        retry++;
        if(SUCCESS == trng_configuration()) {
            if(SUCCESS == trng_ready_check()) {
                /* read the valid data before disabling the TRNG */
                random_data = trng_get_true_random_data();
                result = (uint16_t)(random_data & 0xFFFF); /* low 16 bits of the random data */
                random_status = SUCCESS;
            }

            trng_disable();
        }
    }

    if(ERROR == random_status) {
        /* avoid identical fixed delays when TRNG acquisition fails */
        result = (uint16_t)(SysTick->VAL & 0xFFFFU);
    }

    /* restore the TRNG peripheral clock enable state */
    if(0U == trng_clk_en_bak) {
        rcu_periph_clock_disable(RCU_TRNG);
    }
    /* restore the CK48M clock source to its original setting */
    rcu_ck48m_clock_config(ck48m_src_bak);
    /* turn IRC48M off again if it was originally disabled */
    if(0U == irc48m_en_bak) {
        rcu_osc_off(RCU_IRC48M);
    }

    return result;
}

```

2.10. Core

关于 Cortex-M7 的限制，请参考"Cortex-M7 (AT610) and Cortex-M7 with FPU (AT611) Software Developer Errata Notice"。该文档可在 ARM 官方网站下载。

3. 版本历史

表 3-1. 版本历史

版本号.	说明	日期
1.0	首次发布	2026 年 10 月 27 日

Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the "Product"), is owned by the Company according to the laws of the People's Republic of China and other applicable laws. The Company reserves all rights under such laws and no Intellectual Property Rights are transferred (either wholly or partially) or licensed by the Company (either expressly or impliedly) herein. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

To the maximum extent permitted by applicable law, the Company makes no representations or warranties of any kind, express or implied, with regard to the merchantability and the fitness for a particular purpose of the Product, nor does the Company assume any liability arising out of the application or use of any Product. Any information provided in this document is provided only for reference purposes. It is the sole responsibility of the user of this document to determine whether the Product is suitable and fit for its applications and products planned, and properly design, program, and test the functionality and safety of its applications and products planned using the Product. The Product is designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only, and the Product is not designed or intended for use in (i) safety critical applications such as weapons systems, nuclear facilities, atomic energy controller, combustion controller, aeronautic or aerospace applications, traffic signal instruments, pollution control or hazardous substance management; (ii) life-support systems, other medical equipment or systems (including life support equipment and surgical implants); (iii) automotive applications or environments, including but not limited to applications for active and passive safety of automobiles (regardless of front market or aftermarket), for example, EPS, braking, ADAS (camera/fusion), EMS, TCU, BMS, BSG, TPMS, Airbag, Suspension, DMS, ICMS, Domain, ESC, DCDC, e-clutch, advanced-lighting, etc.. Automobile herein means a vehicle propelled by a self-contained motor, engine or the like, such as, without limitation, cars, trucks, motorcycles, electric cars, and other transportation devices; and/or (iv) other uses where the failure of the device or the Product can reasonably be expected to result in personal injury, death, or severe property or environmental damage (collectively "Unintended Uses"). Customers shall take any and all actions to ensure the Product meets the applicable laws and regulations. The Company is not liable for, in whole or in part, and customers shall hereby release the Company as well as its suppliers and/or distributors from, any claim, damage, or other liability arising from or related to all Unintended Uses of the Product. Customers shall indemnify and hold the Company, and its officers, employees, subsidiaries, affiliates as well as its suppliers and/or distributors harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of the Product.

Information in this document is provided solely in connection with the Product. The Company reserves the right to make changes, corrections, modifications or improvements to this document and the Product described herein at any time without notice. The Company shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. Information in this document supersedes and replaces information previously supplied in any prior versions of this document.