

# GX1832

# GX1833

# 驱动程序

# 用户手册

Rev. 1.1  
2025/02/18

# 修订历史

版本	日期	作者	描述
1.1	2025/02/18	zhengk l	增加 GX1833 专用函数
1.0	2023/04/11	zhengk l	初稿

# 概述

中科银河芯为客户提供了数字式温度传感器 GX1832 的通用驱动程序。该驱动程序将芯片所有功能封装成库,隐藏了从命令格式到基础时序的全部逻辑实现。客户无需专门学习通信协议,直接以函数调用方式即可轻松使用 GX1832,从而大幅缩短项目开发周期。

出于可扩展性考虑,中科银河芯将硬件驱动自上而下划分为三个抽象层级:

- 硬件抽象层:封装了 GX1832 的命令格式;
- 基础协议层:封装了 1-Wire 协议的基础时序;
- 端口映射层:封装了 GPIO 到 DQ 的端口映射与控制。

中科银河芯提供的驱动程序由三个源文件和四个头文件组成,其中三个源文件分别对应了三个抽象层级。具体文件组织结构如下表 1 所示:

表 1. 驱动程序的文件组织结构

文件名	抽象层级	描述
/src/gx1832_driver.c /inc/gx1832_driver.h	硬件抽象层	包括搜索、匹配、转换、读写暂存器等操作
/src/gx1832_onewire.c /inc/gx1832_onewire.h	基础协议层	包括初始化序列、读时序、写时序等操作
/src/gx1832_gpio.c /inc/gx1832_gpio.h	端口映射层	包括端口配置、拉低、释放、采样等操作
/inc/gx1832.h	-	头文件汇总

中科银河芯提供的驱动程序中,端口映射层是基于意法半导体的 ARM STM32F103 实现的。如果采用其他单片机型号,请参照章节 4 进行相关函数覆写。

中科银河芯提供的驱动程序中,变量采用了 C99 标准中的通用数据类型 (stdint.h),因此客户需要在 Keil 工程的 C/C++编译选项中勾选 “C99 Mode”。

GX1832 与 GX1833 共用绝大部分函数,唯一不同的两个函数位于 /src/gx1832\_driver,并通过函数名予以区分: x\_read\_scratchpad 和 x\_generate\_crc8 。

---

---

# 示例

中科银河芯在此提供读取 GX1832x0 测温输出的示例。

```
1 // Step-1 : declare signed variables
2 int16_t temp_hex; // temperature in signed hexadecimal
3 float   temp_cel; // temperature in Celsius degree
4
5 // Step-2 : configure the GPIO (performed only once at power-up)
6 gx1832_gpio_configure();
7
8 // Step-3 : address the specified GX1832
9 gx1832_match(0x4EFFFFFFF029);
10
11 // Step-4 : start temperature conversion
12 gx1832_convert();
13
14 // Step-5 : wait at least 50ms
15
16 // Step-6 : address the specified GX1832
17 gx1832_match(0x4EFFFFFFF029);
18
19 // Step-7 : read temperature result
20 temp_hex = gx1832_read_temperature();
21
22 // Step-8 : calculate the temperature in Celsius degree
23 temp_cel = temp_hex * 0.0625;
```

GX1832 测温输出保存在十六位温度寄存器中，高位已自动填充符号位。客户无需执行任何原码补码转换或移位操作，直接使用十六位有符号整型变量保存函数返回数即可。其中，返回值每个 LSB 代表 0.0625℃。

例如：十六进制 0xFE71 的十进制数值为 -399，代表温度值 -24.9375℃。

# 库说明

表 2. 硬件抽象层的函数原型

函数名	参数	返回值	描述
gx1832_search	RID 数组名	搜索芯片数	执行搜索算法
gx1832_alarm	RID 数组名	搜索芯片数	执行搜索算法（仅报警芯片参与）
gx1832_read	-	RID	读出芯片的 RID
gx1832_match	RID	完成标志	寻址 RID 相匹配的芯片
gx1832_skip	-	完成标志	跳过寻址，并选择总线上所有芯片
gx1832_convert	-	-	启动温度转换
gx1832_read_temperature	-	温度码	读出温度码
gx1832_read_scratchpad	-	暂存器数据	读取暂存器
gx1832_write_scratchpad	暂存器数据	-	修改暂存器
gx1832_generate_crc8	通信数据包	校验码	生成通信数据包的校验码
gx1833_read_scratchpad	-	暂存器数据	读取暂存器
gx1833_generate_crc8	通信数据包	校验码	生成通信数据包的校验码

表 3. 基础协议层的函数原型

函数名	参数	返回值	描述
gx1832_onewire_reset	-	检测结果	发送复位脉冲，并检测响应脉冲
gx1832_onewire_read_bit	-	数据位	启动读时隙，获取一位数据
gx1832_onewire_write_bit	数据位	-	启动写时隙，发送一位数据
gx1832_onewire_read_byte	-	数据字节	连续获取一个字节
gx1832_onewire_write_byte	数据字节	-	连续发送一个字节

表 4. 端口映射层的函数原型

函数名	参数	返回值	描述
gx1832_gpio_configure	-	-	配置端口为开漏输出
gx1832_gpio_pulldown	-	-	拉低总线
gx1832_gpio_release	-	-	释放总线
gx1832_gpio_sample	-	采样结果	采样总线
gx1832_gpio_delay	延时量	-	延时指定微秒

---

---

## 函数覆写

如果客户采用 ARM STM32F103 以外的单片机类型, 仅需要对端口映射层的函数进行覆写, 而无需修改任何其他部分。由于 1-Wire 协议对延时较为敏感, 尤其在长线或多点应用中, 因此建议使用示波器或逻辑分析仪, 将延时函数的准确度调整至 $\pm 10\%$ 以内。

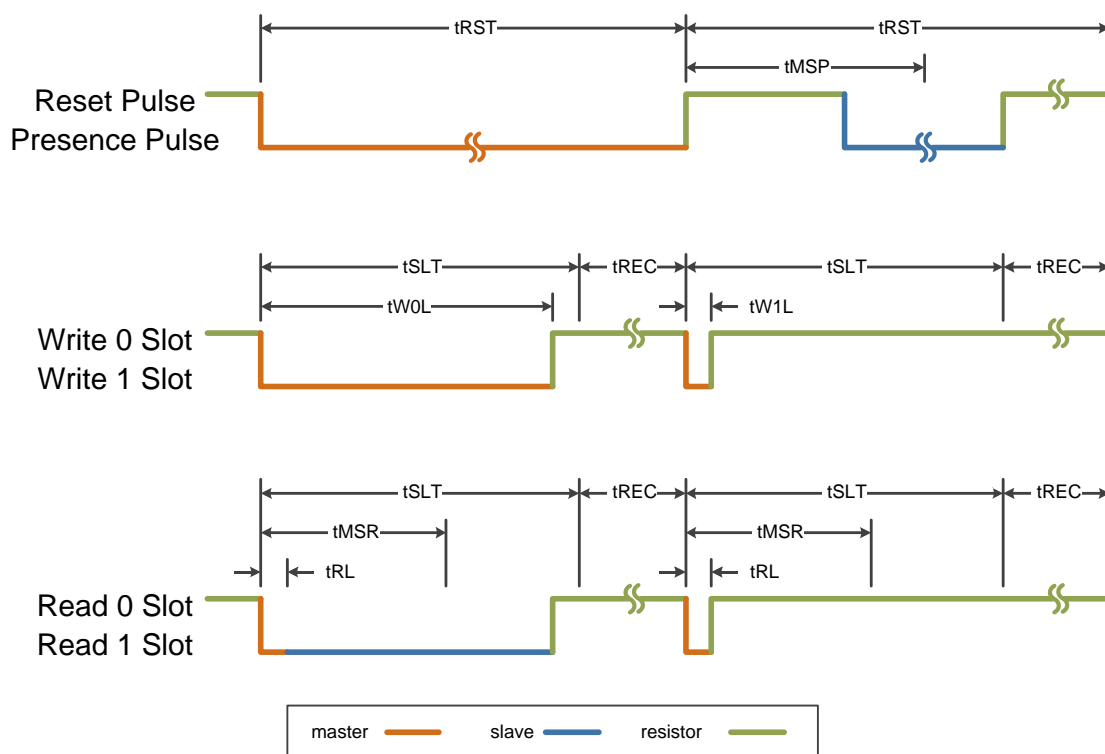
```
1 void gx1832_gpio_configure (void)
2 {
3     // Step-1 : enable the GPIO
4
5     // Step-2 : configure the GPIO as open-drain output
6
7     // Step-3 : release the 1-wire bus
8 }
9
10 void gx1832_gpio_pulldown (void)
11 {
12     // pull down the 1-wire bus
13 }
14
15 void gx1832_gpio_release (void)
16 {
17     // release the 1-wire bus
18 }
19
20 uint8_t gx1832_gpio_sample (void)
21 {
22     // sample the 1-wire bus and save it to 'rxid'
23     return rxid;
24 }
25
26 void gx1832_gpio_delay (uint16_t us)
27 {
28     // delay the specified time (in microsecond)
29 }
```

# 通信速率

中科银河芯提供的驱动程序目前通信速率约为 13kpbs。如果客户需要其他通信速率，可以修改基础协议层的时序定义，位于头文件（/inc/gx1832\_1wire.h）中。

**表 5. 单总线协议的基础时序**

时序	符号	最小值	建议值	最大值
Time Slot Duration	tSLT	65us	70us	-
Recovery Time	tREC	1us	5us	-
Reset Time	tRST	480us	500us	640us
Presence-Detect Sample Time	tMSP	60us	70us	75us
Write-Zero Low Time	tW0L	60us	70us	120us
Write-One Low Time	tW1L	1us	5us	15us
Read Low Time	tRL	1us	5us	15us
Read Sample Time	tMSR	tRL	15us	15us



**图 1. 单总线的基础时序示意图**