玄铁安全系统(OPTEE) C库用户手册

文档编号 STG30004

版权声明

Notice

The purchased products, services and features are stipulated by the contract made between T-HEAD and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

商标申明

平头哥的LOGO和其它所有商标归平头哥半导体有限公司所有，所有其它产品或服务名称归其所有者拥有。

注意

您购买的产品、服务或特性等应受平头哥商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，平头哥对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

平头哥半导体有限公司 T-HEAD Semiconductor Co.,LTD

地址:杭州市余杭区向往街1122号欧美金融城(EFC)英国中心西楼T6 43层 邮编: 311121

修订历史

日期	版本	说明	作者
20220516	v0.1	初版	柏强

1.引言

1.1 面向对象

该文档的面向对象是安全应用的开发者。默认开发者都熟悉linux开发环境。

1.2 编写目的

该文档介绍了TEE-OS向TA开发者提供的C库函数。开发者可以结合自己的开发需求，调用对应的函数接口。

1.3 License 申明

所有的C库的实现都是基于BSD license

2. 接口列表

序号	接口	说明

1	void *malloc(size_t size);	**说明：** 分配所需的内存空间，并返回一个指向它的指针 **参数：** • size -- 内存块的大小，以字节为单位。 **返回值：** 如果成功，该函数返回一个指针 ，指向已分配大小的内存。否则返回 NULL
2	void *calloc(size_t nmemb, size_t size);	**说明：** 分配所需的内存空间，并返回一个指向它的指针，且设置分配的内存为零 **参数：** • nitems -- 要被分配的元素个数。 • size -- 元素的大小。 **返回值：** 如果成功，该函数返回一个指针 ，指向已分配大小的内存。否则返回 NULL
3	void *realloc(void *ptr, size_t size);	**说明：** 重新分配malloc和relloc所分配的内存大小 **参数：** • ptr-- 如果该指针为空，则分配一块新的内存。否则该指针指向之前分配的内存的内存 • size -- 新内存块的大小。 **返回值：** 如果成功，该函数返回一个指针 ，指向已分配大小的内存。否则返回 NULL

4	void free(void *ptr)	**说明：** 释放之前分配的内存块 **参数：** • ptr-- 指向要分配的内存块的指针 **返回值：** 无
5	int printf(const char *fmt, ...)	**说明：** 发送格式化数据到标准输出 **参数：** • fmt-- 要被写入标准输出的文本 **返回值：** 如果成功，则返回写入的字符总数，否则返回一个负数
6	int sprintf(char *str, const char *fmt, ...)	**说明：** 发送格式化数据到str指向的存储区 **参数：** • str-- 指向格式化数据输出区域的指针 • fmt-- 要被写入标准输出的文本 **返回值：** 如果成功，则返回写入的字符总数，否则返回一个负数
7	int snprintf(char *str, size_t size, const char *fmt, ...)	**说明：** 将可变参数“...”按照format的格式格式化为字符串，然后再将其拷贝至str中。 **参数：** • str-- 指向格式化数据输出区域的指针 • size -- 要拷贝的字节数 • fmt-- 要被写入标准输出的文本 **返回值：** 如果成功，则返回写入的字符总数，否则返回一个负数

8	int puts(const char *str);	**说明：** 把一个字符串写入到标准输出 stdout，直到空字符，但不包括空字符 **参数：** • **str** -- 这是要被写入的 C 字符串。 **返回值：** 如果成功，则返回写入的字符总数，否则返回EOF
9	int putchar(int c);	**说明：** 输出一个字符 **参数：** • c-- 这是要被写入的 字符 **返回值：** 如果成功，则返回写入的字符总数，否则返回EOF
10	void qsort(void *aa, size_t n, size_t es, int (*cmp)(const void *, const void *));	**说明：** 对数组进行排序 **参数：** • base -- 指向要排序的数组的第一个元素的指针。 • nitems -- 由 base 指向的数组中元素的个数。 • size -- 数组中每个元素的大小，以字节为单位。 • compar -- 用来比较两个元素的函数。 **返回值：** 无

11	void abort(void) __noreturn;	**说明：** 中止程序执行，直接从调用的地方跳出 **参数：** 无 **返回值：** 无
12	int abs(int i);	**说明：** 取绝对值 **参数：** • i—— 要被取绝对值的数 **返回值：** i的绝对值
13	int rand(void);	**说明：** 返回一个随机数 **参数：** 无 **返回值：** 随机值

14	unsigned long strtoul (const char *s, char **ptr, int base);	**说明：** 把参数s所指向的字符串根据给定的 base 转换为一个无符号长整数（类型为 unsigned long int 型） **参数：** • s -- 要转换为无符号长整数的字符串。 • endptr -- 对类型为 char* 的对象的引用，其值由函数设置为 str 中数值后的下一个字符。 • base -- 基数，必须介于 2 和 36（包含）之间，或者是特殊值 0。 **返回值：** 该函数返回转换后的长整数，如果没有执行有效的转换，则返回一个零值
15	void *memcpy(void *__restrict s1, const void *__restrict s2, size_t n);	**说明：** 从s2指针的区域拷贝n个字节到s1指向的区域 **参数：** • s1-- 指向用于存储复制内容的目标数组，类型强制转换为 void* 指针。 • s2-- 指向要复制的数据源，类型强制转换为 void* 指针。 • n -- 要被复制的字节数。 **返回值：** 该函数返回一个指向目标存储区 s1的指针。

16	void *memmove(void *s1, const void *s2, size_t n);	**说明：** 从s2指针的区域拷贝n个字节到s1指向的区域 **参数：** • s1-- 指向用于存储复制内容的目标数组，类型强制转换为 void* 指针。 • s2-- 指向要复制的数据源，类型强制转换为 void* 指针。 • n -- 要被复制的字节数。 **返回值：** 该函数返回一个指向目标存储区 s1的指针。
17	int memcmp(const void *s1, const void *s2, size_t n);	**说明：** 比较存储区 s1和存储区 s2的前 n 个字节。 **参数：** • s1-- 指向内存块的指针。 • s2-- 指向内存块的指针。 • n -- 要被比较的字节数。 **返回值：** • 如果返回值 < 0，则表示 s1 小于 s2。 • 如果返回值 > 0，则表示 s1 大于 s2。 • 如果返回值 = 0，则表示 s1 等于 s2。

	void *memset(void *s, int c, size_t n);	**说明：** 填充n个字符c到s指向的区域 **参数：** • s –– 指向要填充的内存块。 • c –– 要被设置的值。该值以 int 形式传递，但是函数在填充内存块时是使用该值的无符号字符形式。 • n –– 要被设置为该值的字符数。 **返回值：** 无
18	int strcmp(const char *s1, const char *s2);	**说明：** 比较字符串s1和s2。 **参数：** • s1–– 指向要比较的字符串1 • s2–– 指向要比较的字符串2 **返回值：** • 如果返回值 < 0，则表示 s1 小于 s2。 • 如果返回值 > 0，则表示 s1 大于 s2。 • 如果返回值 = 0，则表示 s1 等于 s2。
19	int strncmp(const char *s1, const char *s2, size_t n);	**说明：** 比较字符串s1和s2的前n个字节 **参数：** • s1–– 指向要比较的字符串1 • s2–– 指向要比较的字符串2 • n–– 要比较的字节数 **返回值：** • 如果返回值 < 0，则表示 s1 小于 s2。 • 如果返回值 > 0，则表示 s1 大于 s2。 • 如果返回值 = 0，则表示 s1 等于 s2。

20	size_t strlen(const char *s);	**说明：** 统计字符串s的长度 **参数：** • s1-- 指向要统计的字符串 **返回值：** 字符串的长度
21	size_t strnlen(const char *s, size_t n);	**说明：** 统计字符串s的长度，如果长度大于n，则返回n **参数：** • s-- 指向要统计的字符串 • n-- 字符串最大长度 **返回值：** 字符串的长度，如果长度大于n，则返回n
22	char *strdup(const char *s);	**说明：** 复制一个字符串 **参数：** • s-- 要复制的字符串 **返回值：** 字符串被复制的目标地址
23	char *strndup(const char *s, size_t n);	**说明：** 带长度限制的复制字符串 **参数：** • s-- 要复制的字符串 • n-- 复制的最大长度 **返回值：** 字符串被复制的目标地址

24	char *strchr(const char *s, int c);	**说明：** 在s所指向的缓冲区搜索第一次出现字符c的位置 **参数：** • s-- 被搜索的字符串 • c-- 要搜索的字符 **返回值：** 第一次出现字符c的位置
25	char *strstr(const char *big, const char *little);	**说明：** 在big所指向的缓冲区搜索第一次出现字符串little的位置 **参数：** • big-- 被搜索的字符串 • little-- 要搜索的字符串 **返回值：** 第一次出现字符串little的位置
26	char *strcpy(char *dest, const char *src);	**说明：** 把src指向的字符串拷贝到dest指针的地址中 **参数：** • dest-- 目标地址 • src-- 指向被拷贝的字符串的指针 **返回值：** 指向最终的目标字符串 dest 的指针

27	char *strncpy(char *dest, const char *src, size_t n);	**说明：** 把src指向的字符串拷贝n个字节到dest指针的地址中 **参数：** • dest-- 目标地址 • src-- 指向被拷贝的字符串的指针 • n -- 要拷贝的字节数 **返回值：** 指向最终的目标字符串 dest 的指针
28	char *strrchr(const char *s, int i);	**说明：** 在s所指向的缓冲区搜索第一次出现字符i的位置 **参数：** • s-- 被搜索的字符串 • i-- 要搜索的字符 **返回值：** 第一次出现字符i的位置
29	void *memchr(const void *buf, int c, size_t length);	**说明：** 在buf所指向的缓冲区的前length个字节搜索第一次出现字符c的位置 **参数：** • buf-- 被搜索的字符串 • c-- 要搜索的字符 • length -- buf所指向的缓冲区，要搜索的字节数 **返回值：** 第一次出现字符c的位置

30	int bcmp(const void *s1, const void *s2, size_t n);	**说明：** 比较S1和S2所指向的字符串的前n字节是否相等 **参数：** • s1-- 指向字符串1的指针 • s1-- 指向字符串2的指针 • n-- 要比较的字节数 **返回值：** 相等返回0，否则返回非零
31	size_t strlcpy(char *dst, const char *src, size_t size);	**说明：** 把src指向的字符串拷贝到dst指针的地址中 **参数：** • dst-- 目标地址 • src-- 指向被拷贝的字符串的指针 • size-- 需要copy的size **返回值：** 拷贝的size，不包含NULL
32	size_t strlcat(char *dst, const char *src, size_t size);	**说明：** 把src指向的字符串附加到dst指针的地址中 **参数：** • dst-- 目标地址 • src-- 指向被拷贝的字符串的指针 • size-- dest的buffer size **返回值：** src字符串的大小