

TinyFrame_Interface

Hare_breath& Heart rhythm

1. 应用项目

针对不同的应用项目，列出所有与TF帧相关的消息，供用户参考并完成解析，对于文档所出现的消息类，以及消息数据位，是在对应实际项目所配备的。

1.1. 呼吸+心率检测项目

消息类型：报告相位测试结果 0x0A13

消息类型为0x0A13，仅支持单向数据传输模式。

雷达发送数据给上位机					
格式	字节数	基本类型	帧结构	示例帧	帧含义
SOF	1 byte	uint8	起始帧	01	用于输出总相位、心跳相位、呼吸相位结果。
ID	2 byte	uint16	帧 ID	00 00	
LEN	2 byte	uint16	数据帧长度	00 04	
TYPE	2 byte	uint16	帧类型	0A 13	
HEAD_CHK SUM	1 byte	uint8	头校验和	\	
DATA	4 byte	float	[total phase]	\	
DATA	4 byte	float	[breath phase]	\	
DATA	4 byte	float	[heart phase]	\	
DATA_CHK SUM	1 byte	uint8	数据校验和	\	

消息类型：报告呼吸率检测项目测试结果 0x0A14

消息类型为 0x0A14，支持单向数据传输模式。

雷达发送数据给上位机					
格式	字节数	基本类型	帧结构	示例帧	帧含义
SOF	1 byte	uint8	起始帧	01	用于报告呼吸速率测试结果。
ID	2 byte	uint16	帧 ID	00 00	
LEN	2 byte	uint16	数据帧长度	00 04	
TYPE	2 byte	uint16	帧类型	0A 14	
HEAD_CHK SUM	1 byte	uint8	头校验和	\	
DATA	4 byte	float	[rate]	\	
DATA_CHK SUM	1 byte	uint8	数据校验和	\	

消息类型：报告心跳速率检测结果 0x0A15

消息类型为 0x0A15，支持单向数据传输模式。

雷达发送数据给上位机					
格式	字节数	基本类型	帧结构	示例帧	帧含义
SOF	1 byte	uint8	起始帧	01	用于报告心跳相位测试结果。
ID	2 byte	uint16	帧 ID	00 00	
LEN	2 byte	uint16	数据帧长度	00 04	
TYPE	2 byte	uint16	帧类型	0A 15	
HEAD_CKSUM	1 byte	uint8	头校验和	\	
DATA	4 byte	float	[rate]	\	
DATA_CKSUM	1 byte	uint8	数据校验和	\	

消息类型：报告检测目标距离 0x0A16

消息类型为 0x0A16，支持单向数据传输模式。

雷达发送数据给上位机					
格式	字节数	基本类型	帧结构	示例帧	帧含义
SOF	1 byte	uint8	起始帧	01	用于报告检测距离。
ID	2 byte	uint16	帧 ID	00 00	
LEN	2 byte	uint16	数据帧长度	00 04	
TYPE	2 byte	uint16	帧类型	0A 16	
HEAD_CKSUM	1 byte	uint8	头校验和	\	
DATA	4 byte	uint 32	[flag]	\	
DATA	4 byte	float	[range]	\	
DATA_CKSUM	1 byte	uint8	数据校验和	\	

注：标志为1时，输出距离（单位：cm）

标志为0时，不输出距离

A、以下是DATA位的数据转换：

转换成float：例如[rate]位为0x66、0x66、0xA2、0x41，先拼成uint32位整形，由于TF帧Data位小端序，所以值为0x41A26666，然后进行float类型强转，最终结果为：20.3。

```
1. int main(void)
2. {
3.     unsigned int param = 0x41A26666;
4.     float res = *(float *)&param;
5.
6.     printf("data: %f\n", res);
7.     return 0;
8. }
```

B、以下是每个CKSUM的解析：

HEAD_CKSUM：TF帧头校验和【从第一个字节开始到HEAD_CKSUM位的上一个字节】

DATA_CKSUM：TF数据验和【DATA的第一个到DATA_CKSUM位的上一个字节】

其中计算CKSUM的方法c代码如下所示：

```
1. unsigned char getCksum(unsigned char *data, unsigned char len)
2. {
3.     unsigned char ret = 0;
4.
5.     for (int i = 0; i < len; i++)
```

```
6.         ret = ret ^ data[i];
7.
8.         ret = ~ret;
9.
10.        return ret;
11.    }
12.
```

2. 编程接口

2.1. 编码TF消息

`void tinyFrameTx(TF_TYPE type, uint8 *data, TF_LEN len);`

其中type为发送数据类型，uint16类型。

Uint8* data是发送数据的地址。

Len 为发送数据的长度，uint16类型。

2.2. 解码 TF 消息

`TinyFrameRx tinyFrameRx(void);`

成功接收消息后，接收的数据返回到一个TinyFrameRx类型的变量。

2.3. 示例代码

如果想要解析TF 帧数据的demo（包含 Linux 环境与Keil μ Vision5 环境下的C语言demo、Python语言demo），可以直接与销售沟通获得。