HYPERSEN TECHNOLOGIES CO., LTD.

HPS-FT-EN2000-IO

六维力传感器 Ethernet 适配器

概述

1 产品描述

HPS-FT-EN2000-IO 适配器用于将六维力传感器的 RS485 接口转换为以太网百兆接口，同时支持更丰富和更简单的命令设置。数据输出频率最高可达 2000HZ。

1.1 外形尺寸

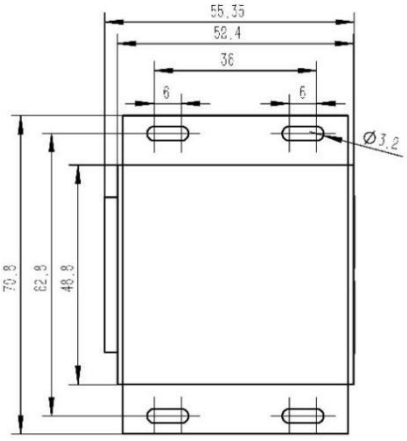

图 1. 适配器俯视图

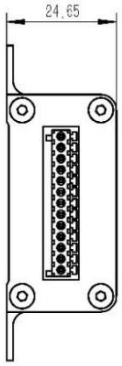

图 2. 适配器左视图

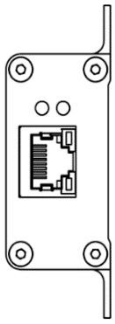

图 3. 适配器右视图

表 1. Ethernet 适配器接线定义

端口号	信号名称	信号种类	描述
1	适配器电源正极	VCC	连接到 DC +12V ~ +24V
2	适配器电源负极	GND	连接到电源地

3	适配器屏蔽地	SHIELD	连接到机壳地
4	CH1 传感器电源正极	VCC	连接传感器电源正极(红色电缆)
5	CH1 传感器电源负极	GND	连接传感器电源负极(黑色电缆)
6	CH1 传感器 485A	Digital	连接传感器 485 A 端(白色电缆)
7	CH1 传感器 485B	Digital	连接传感器 485 B 端(绿色电缆)
8	CH2 传感器电源正极	VCC	连接传感器电源正极(红色电缆)
9	CH2 传感器电源负极	GND	连接传感器电源负极(黑色电缆)
10	CH2 传感器 485A	Digital	连接传感器 485 A 端(白色电缆)
11	CH2 传感器 485B	Digital	连接传感器 485 B 端(绿色电缆)
12*1	光耦输出 OUT	Digital	光耦输出端
13*2	光耦 COM 端	GND	光耦 COM 端

注: *1 *2

报警信号的输出端(端口号 12)是无电源 NPN 输出类型.需要外部提供电流驱动的 GPIO 来驱动。COM 端(端口号 13)和 GND 的信号地没有连接在一起,用户可自行选择外部连接在一起或者隔离不连接在一起。建议 RL 和 VCC 的选取,使得导通后的电流 IC(集电极电流)控制在 5mA-40mA 之间。输出电流大于 40mA 的情况建议用户在后端自行增加一级功率驱动，避免电流太大导致设备故障。

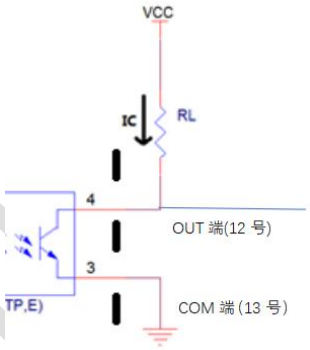

表 2、LED 指示灯功能

上方指示灯	常亮	通道 1 传感器通讯正常
	不亮	通道 1 传感器通讯异常
下方指示灯	常亮	通道 2 传感器通讯正常
	不亮	通道 2 传感器通讯异常

控制界面

2.1 Ethernet 总线通讯协议

Ethernet 适配器使用使用 UDP 或 TCP 协议通讯，默认为 UDP 协议，可以通过切换设配器内部的 2 号拨码开关选择 TCP 协议。默认 IP 为 192.168.1.100，掩码为 255.255.255.0，网关为 192.168.1.1，端口号为 8080，用户可以通过命令修改 IP、掩码、网关和端口号（端口号 50000 为调试端口，用户无法使用）。将适配器内部的 1 号拨码开关拨到不同于默认位置，则适配器每次重启所有参数都会被初始化为默认参数。

2.2 数据帧格式:

Ethernet 适配器有两种数据解析模式 1、二进制模式 2、ASCII 模式（适用于 Ethernet 适配器固件版本为 1.4 及以上版本），出厂默认为二进制模式，用户可以通过命令切换模式。

2.2.1 二进制数据帧格式

表 3. 传感器数据帧格式

帧头		有效数据长度	设备地址	保留字节	命令	内容	CRC 低字节	CRC 高字节	帧尾	
Byte0	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6~N	ByteN+1	ByteN+2	ByteN+3	ByteN+4
0xF6	0x6F	1 字节	0x00 (默认)	0x00	1 字节	0~255 字节	1 字节	1 字节	0x6F	0xF6

注： 1. CRC 校验为 CRC16-CCITT-FALSE，附录中附有 CRC16 C 语言实现代码；CRC 校验范围为有效数据范围（设备地址~内容）

2. 设备地址 0 用于配置通道 1 的传感器，设备地址 1 用于配置通道 2 的传感器，设备地址 2 用于同时配置两个通道的传感器

2.2.2 命令列表

表 4. 传感器命令列表

命令	命令字节	描述
读取设备 ID	0x01	读取设备的 ID 号: 0x46FE
连续测量	0x02	连续输出测量数据
停止连续测量	0x03	停止连续测量
单次测量	0x04	单次输出测量数据
保存用户设置	0x09	将用户设置保存到 Flash 中
获取传感器版本信息	0x0A	获取传感器的版本信息
零点校正	0x0B	执行传感器零点校正
获取传感器序列号	0x10	获取传感器序列号
使能卡尔曼滤波	0x11	使能卡尔曼滤波
设置卡尔曼滤波参数	0x12	设置卡尔曼滤波参数
工具坐标系转换	0x13	工具坐标系转换
获取适配器固件版本号	0x14	获取适配器固件版本号
初始化传感器	0x15	该命令用于枚举和初始化传感器
激活端口 2 传感器	0x16	该命令用于激活通道 2 的传感器，默认只激活通道 1
获取传感器状态码	0x17	获取传感器状态码，用于确定传感器和适配器是否通讯正常
设置低通滤波参数	0x18	设置低通滤波参数
设置适配器的 IP 地址	0x19	设置适配器的 IP 地址
设置适配器的掩码	0x1A	设置适配器的掩码
设置适配器的网关	0x1B	设置适配器的网关
设置适配器的端口号	0x1C	设置适配器的端口号
设置适配器解析模式	0x1E	设置适配器的解析模式，出厂默认为二进制模式
设置中值滤波深度	0x20	设置中值滤波深度，一共有 0-64 个深度
清除 IO 报警信号	0x23	清除传感器的 IO 报警信号
设置 IO 报警信号的阈值	0x24	设置 IO 报警信号的阈值

使能传感器 IO 报警	0x25	使能传感器 IO 报警，出厂默认为不使能
获取触发 IO 报警的轴信息	0x26	获取触发 IO 报警的轴,0~5 分别代表 Fx、Fy、Fz、Mx、My、Mz
设置 IO 报警端口为常开或常闭合模式	0x27	默认为常开模式，即报警信号没有触发时，OUT 端口和 COM 端口是断开的
手动触发传感器 IO 报警	0x28	该命令用于手动触发 IO 报警
获取过载次数信息	0xD4	获取过载次数信息
获取过载峰值信息	0xD8	获取过载峰值信息

命令#1 获取传感器的设备 ID 号

该命令可获取传感器的设备 ID 号（0x46FE）。

表 5. 获取传感器的设备 ID 号命令

帧头		有效数据长度	设备地址	保留字节	命令	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte3	Byte4	Byte5	Byte6	Byte7
0xF6	0x6F	0x03	0x00	0x00	0x01	0xBD	0xDC	0x6F	0xF6

返回数据:

表 6. 获取传感器的设备 ID 号命令的返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x05	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x01	命令字节
6	0xFE	设备 ID 号低字节
7	0x46	设备 ID 号高字节
8	0xF0	CRC16-CCITT 校验低字节
9	0x3E	CRC16-CCITT 校验高字节
10	0x6F	帧尾第一字节
11	0xF6	帧尾第二字节

命令#2 单次测量

此命令将启动传感器进行一次单次测量。

表 7. 单次测量命令

帧头		有效数据长度	设备地址	保留字节	命令	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9
0xF6	0x6F	0x03	0x00	0x00	0x04	0x18	0x8C	0x6F	0xF6

命令#3 连续测量

此命令将启动传感器进行连续测量。

表 8. 连续测量命令

帧头		有效数据长度	设备地址	保留字节	命令	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte3	Byte4	Byte5	Byte6	Byte7
0xF6	0x6F	0x03	0x00	0x00	0x02	0xDE	0xEC	0x6F	0xF6

单次/连续测量命令返回数据:

表 9. 单次/连续测量命令的返回数据(没有激活传感器通道 2)

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x1B	有效数据长度
3	0x00	设备地址，在激活通道 2 传感器时，该地址为 0x02
4	0x00/0xFF/0xFE	异常数据指示，0 表示数据正常、0xFF 表示数据出现异常、0xFE 表示传感器当前处于过载状态下
5	0x04/0x02	命令字节
6~9	……	Fx 的低字节~高字节的整型补码，该数值/1000=Fx （单位：N）
10~13	……	Fy 的低字节~高字节的整型补码，该数值/1000=Fy （单位：N）
14~17	……	Fz 的低字节~高字节的整型补码，该数值/1000=Fz （单位：N）
18~21	……	Mx 的低字节~高字节的整型补码，该数值/1000=Mx （单位：Nm）
22~25	……	My 的低字节~高字节的整型补码，该数值/1000=My （单位：Nm）
26~29	……	Mz 的低字节~高字节的整型补码，该数值/1000=Mz （单位：Nm）
30	……	CRC16-CCITT 校验低字节
31	……	CRC16-CCITT 校验高字节
32	0x6F	帧尾第一字节
33	0xF6	帧尾第二字节

数据解析例：

以下为连续测量模式下的一帧测量数据:

F6 6F 1B 00 00 02 16 FF FF FF 01 FA FF FF EF 02 00 00 06 00 00 00 0A 00 00 00 0F 00 00 00 6F 58 6F F6

解析：

0xF6, 0x6F: 帧头

0x1B: 有效数据长度

0x00: 设备地址

0x00: 数据正常

0x02: 连续测量命令字节

0x16 0xFF 0xFF 0xFF: Fx 数据，0xFFFFFF16 → -234，对应 X 轴的力为-0.234N

0x01 0xFA 0xFF 0xFF: Fy 数据 0xFFFFFA01 → -1535，对应 Y 轴的力为-1.535N

0xEF 0x02 0x00 0x00: Fz 数据 0x000002EF → 751，对应 Z 轴的力为 0.751N

0x06 0x00 0x00 0x00: Mx 数据 0x00000006 → 6，对应 X 轴的力矩为 0.006Nm

0x0A 0x00 0x00 0x00: My 数据 0x0000000A → 10，对应 Y 轴的力矩为 0.010Nm

0x0F 0x00 0x00 0x00 Mz 数据 0x0000000F → 15，对应 Z 轴的力矩为 0.015Nm

0x6F: CRC16-CCITT 校验低字节

0x58: CRC16-CCITT 校验高字节

0x6F, 0xF6: 帧尾

表 10. 单次/连续测量命令的返回数据（激活通道 2 传感器）

字节号	数据	描述

0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x3B	有效数据长度
3	0x02	激活通道 2 传感器时，该地址为 0x02
4	0x00/0xFF/0xFE	异常数据指示，0 表示数据正常、0xFF 表示数据出现异常、0xFE 表示传感器当前处于过载状态下
5	0x04/0x02	命令字节
6~9	……	通道 1 传感器的 Fx 的低字节~高字节的整型补码，该数值/1000=Fx（单位：N）
10~13	……	通道 1 传感器的 Fy 的低字节~高字节的整型补码，该数值/1000=Fy（单位：N）
14~17	……	通道 1 传感器的 Fz 的低字节~高字节的整型补码，该数值/1000=Fz（单位：N）
18~21	……	通道 1 传感器的 Mx 的低字节~高字节的整型补码，该数值/1000=Mx（单位：Nm）
22~25	……	通道 1 传感器的 My 的低字节~高字节的整型补码，该数值/1000=My（单位：Nm）
26~29	……	通道 1 传感器的 Mz 的低字节~高字节的整型补码，该数值/1000=Mz（单位：Nm）
30~33	……	4 个字节的通道 1 传感器内部采样次数计数，Ethernet 转接板每次接收到传感器一帧数据会加 1。
34~37	……	通道 2 传感器的 Fx 的低字节~高字节的整型补码，该数值/1000=Fx（单位：N）
38~41	……	通道 2 传感器的 Fy 的低字节~高字节的整型补码，该数值/1000=Fy（单位：N）
42~45	……	通道 2 传感器的 Fz 的低字节~高字节的整型补码，该数值/1000=Fz（单位：N）
45~48	……	通道 2 传感器的 Mx 的低字节~高字节的整型补码，该数值/1000=Mx（单位：Nm）
49~52	……	通道 2 传感器的 My 的低字节~高字节的整型补码，该数值/1000=My（单位：Nm）
53~56	……	通道 2 传感器的 Mz 的低字节~高字节的整型补码，该数值/1000=Mz（单位：Nm）
57~60	……	4 个字节的通道 1 传感器内部采样次数计数，Ethernet 转接板每次接收到传感器一帧数据会加 1。
61	……	CRC16-CCITT 校验低字节
62	……	CRC16-CCITT 校验高字节

注意：

1、传感器在上电后，需要预热一段时间以稳定输出，建议以最高的测量频率工作 10~20 分钟，使内部器件受热平衡后发送零点复位命令，复位完成后再开始使用。

2、通过返回数据包的 byte 4 检查当前的数据是否正常，如果该字节为 0xFF 则传感器可能在实际使用中由于操作不当造成传感器内部结构损坏，此时应该立即停止使用，联系厂家以定位传感器异常原因。如果该字节为 0xFE 则表示传感器当前在过载状态下工作，原则上传感器需要在量程之内使用，工作时力或力矩不能长时间超量程使用，

短时过载不会对传感器造成损害（在过载限度内）。用户在使用时应该确保传感器的静态载荷和动态载荷(机器人在高速运动时会产生较大的加速度，传感器会受到较大的力和力矩)在传感器的量程之内。

命令#4 停止连续测量

此命令将停止传感器的连续测量输出。

表 11. 停止连续测量命令

帧头		有效数据长度	设备地址	保留字节	命令	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9
0xF6	0x6F	0x03	0x00	0x00	0x03	0xFF	0xFC	0x6F	0xF6

返回数据： 无返回数据

命令#5 保存用户设置

此命令可以将用户设置保存到传感器内部的 Flash 中，保存的用户设置断电不会丢失，并将在每一次传感器上电时被自动加载。

表 12. 保存用户设置命令

帧头		有效数据长度	设备地址	保留字节	命令	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9
0xF6	0x6F	0x03	0x00	0x00	0x09	0xB5	0x5D	0x6F	0xF6

返回数据:

表 13. 保存用户设置命令的返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x09	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0x79	CRC16-CCITT 校验低字节
8	0x2E	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

注：该命令执行时间大约为 3s，用户要注意读取返回数据的超时时间。

命令#6 获取传感器版本信息

此命令可用来获取传感器的版本信息。

表 14. 获取传感器版本信息命令

帧头	有效数据长度	设备地址	保留字节	命令	CRC 低字节	CRC 高字节	帧尾

Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte3	Byte4	Byte5	Byte6	Byte7
0xF6	0x6F	0x03	0x00	0x00	0x0A	0xD6	0x6D	0x6F	0xF6

返回数据:

表 15. 获取传感器版本信息命令的返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x0A	命令字节
6	……	固件生成时间，年
7	……	固件生成时间，月
8	……	固件生成时间，日
9	……	主版本号
10	……	小版本号
11	……	修订版本号
12	……	CRC16-CCITT 校验低字节
13	……	CRC16-CCITT 校验高字节
14	0x6F	帧尾第一字节
15	0xF6	帧尾第二字节

注意：若激活通道 2 传感器，则字节 12~17 为通道 2 传感器的版本信息

命令#7 零点复位

此命令用来复位传感器的零点输出，传感器安装完毕后发送该命令即可复位传感器的零点输出。

表 16. 零点复位命令

帧头		有效数据长度	设备地址	保留字节	命令	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte3	Byte4	Byte5	Byte6	Byte7
0xF6	0x6F	0x03	0x00	0x00	0x0B	0xF7	0x7D	0x6F	0xF6

返回数据:

表 17. 零点复位命令的返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x0B	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0x1B	CRC16-CCITT 校验低字节
8	0x48	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节

10	0xF6	帧尾第二字节

注意：设备地址 0x00 用于复位通道 1 传感器； 0x01 地址用于复位通道 2 传感器；0x02 地址用于复位两个通道的传感器。

命令#8 获取传感器序列号

表 18. 获取传感器序列号命令

帧头		有效数据长度	设备地址	保留字节	命令	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9
0xF6	0x6F	0x03	0x00	0x00	0x10	0xAD	0xDE	0x6F	0xF6

表 19. 获取传感器序列号命令的返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x0B	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x10	命令字节
6~13	……	8 bytes 传感器序列号，
14	……	CRC16-CCITT 校验低字节
15	……	CRC16-CCITT 校验高字节
16	0x6F	帧尾第一字节
17	0xF6	帧尾第二字节

注意：若激活通道 2 传感器，则字节 14~21 为通道 2 传感器的序列号。

命令#9 使能卡尔曼滤波

此命令用于使能卡尔曼滤波器。

表 20. 使能卡尔曼滤波命令

帧头		有效数据长度	设备地址	保留字节	命令	使能或禁用滤波	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9	Byte10
0xF6	0x6F	0x04	0x00	0x00	0x11	0x01	0xA3	0xA4	0x6F	0xF6

表 21. 使能卡尔曼滤命令的返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x11	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0xA3	CRC16-CCITT 校验低字节
8	0xA4	CRC16-CCITT 校验高字节

9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

注意： 设备地址 0x00 用于使能通道 1 传感器卡尔曼滤波； 0x01 地址用于使能通道 2 传感器卡尔曼滤波；0x02 地址用于使能两个通道的卡尔曼滤波。

命令#10 设置卡尔曼滤波参数

此命令可用来设置卡尔曼滤波参数

表 22. 设置卡尔曼滤波参数命令

帧头		有效数据长度	设备地址	保留字节	命令	滤波参数	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte6~23	Byte24	Byte25	Byte26	Byte27
0xF6	0x6F	0x15	0x00	0x00	0x12	……	……	……	0x6F	0xF6

表 24.卡尔曼滤波参数

字节号	名称	功能	描述
6	kalman_K	力的新数据的权重	力的新数据的权重，可配置为 1 ~ 100 默认值：20
7~10	kalman_threshold	力的噪声数据的阈值	力的噪声数据的阈值，1000 对应 1N 默认值：250
11~14	num_check	连续超过噪声阈值的力的帧数	连续超过噪声阈值的力的帧数 默认值：3
15	kalman_K	力矩新数据的权重	力矩新数据的权重，可配置为 1 ~ 100 默认值：20
16~19	kalman_threshold	力矩噪声数据的阈值	力矩噪声数据的阈值，1000 对应 1N*M 默认值：2
20~23	num_check	连续超过噪声阈值的力矩的帧数	连续超过噪声阈值的力矩的帧数 默认值：3

注：

1. a. 新数据的权重：适当降低该数值可降低数据噪声，但会降低瞬间响应速度。
   b. 噪声数据的阈值：该值应该略大于当前传感器的背景噪声值。
   c. 连续超过噪声阈值的的帧数：数据值大于噪声数据的阈值的连续帧数量超过该数值时数据有效，适当提高该值可以减小数据的跳动，但会损失一些瞬时响应速度。

2、设备地址 0x00 用于设置通道 1 传感器卡尔曼滤波； 0x01 地址用于设置通道 2 传感器卡尔曼滤波；0x02 地址用于设置两个通道的卡尔曼滤波

例：

以下命令将力的卡尔曼滤波参数设置为: kalman_K = 20, kalman_threshold = 800, num_check=2,
力矩的卡尔曼滤波参数设置为: kalman_K = 20, kalman_threshold = 2, num_check=2.
F6 6F 15 00 00 12 14 20 03 00 00 02 00 00 00 14 02 00 00 00 02 00 00 00 95 5A 6F F6

解析：

0xF6, 0x6F: 帧头
0x15: 有效数据长度
0x00: 设备地址,配置通道 1 传感器

0x00: 保留字节
0x12 : 设置卡尔曼滤波参数命令
0x14: 力的 kalman_K = 20
0x20 0x03 0x00 0x00: 力的 kalman_threshold = 800，对应 0.8N
0x02 0x00 0x00 0x00: 力的 num_check=2
0x14: 力矩的 kalman_K = 20
0x02 0x00 0x00 0x00: 力矩的 kalman_threshold = 2，对应 0.002N*m
0x02 0x00 0x00 0x00: 力矩的 num_check=2
0x95: CRC16-CCITT 校验低字节
0x5A: CRC16-CCITT 校验高字节
0x6F, 0xF6: 帧尾

返回数据:

表 23. 设置卡尔曼滤波参数命令的返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x12	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0xF0	CRC16-CCITT 校验低字节
8	0xF1	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

命令#11 工具坐标系转换

此命令可用于工具坐标系转换

表 24. 工具坐标系转换命令命令

帧头		有效数据长度	设备地址	保留字节	命令	工具坐标系参数	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte6~29	Byte30	Byte31	Byte32	Byte33
0xF6	0x6F	0x1B	0x00	0x00	0x13	……	……	……	0x6F	0xF6

表 25.工具坐标系转换参数参数

字节号	名称	描述
6~9	Rx	工具坐标系围绕 X 轴的旋转角度
10~13	Ry	工具坐标系围绕 Y 轴的旋转角度
14~17	Rz	工具坐标系围绕 Z 轴的旋转角度
18~21	Dx	工具坐标系向 X 轴方向的平移距离
22~25	Dy	工具坐标系朝 Y 轴方向的平移距离
26~29	Dz	工具坐标系朝 Z 轴方向的平移距离

注：1. 坐标系围绕某个轴旋转时，顺时针为正，1000 个单位对应 1 度，旋转顺序为 X 轴->Y 轴->Z 轴。工具坐标系向某个轴方向平移时，1000 个单位对应 1 米。

例：Rx = 90000 表示将工具坐标系围绕 X 轴顺时针旋转 90 度。

Dx = 50 表示将工具坐标系向 X 轴正方向平移 5cm。

2. 地址 0x00 用于设置通道 1 传感器工具坐标系转换； 0x01 地址用于设置通道 2 传感器工具坐标系转换；0x02 地址用于设置两个通道的工具坐标系转换。

以下命令将 X 轴旋转 90 度，Z 轴旋转 90 度：

F6 6F 1B 00 00 13 90 5F 01 00 00 00 00 00 90 5F 01 00 00 00 00 00 00 00 00 00 00 00 00 D4 0B 6F F6

解析：

0xF6, 0x6F:	帧头
0x1B:	有效数据长度
0x00:	设备地址,配置通道 1 传感器
0x00:	保留字节
0x13 :	工具坐标系转换命令
0x90 0x5F 0x01 0x00:	Rx=90000 对应将 X 轴旋转 90 度
0x00 0x00 0x00 0x00:	Ry=0，Y 轴不旋转
0x90 0x5F 0x01 0x00:	Rz=90000,对应将 Z 轴旋转 90 度
0x00 0x00 0x00 0x00:	Dx=0，X 轴不平移
0x00 0x00 0x00 0x00:	Dy=0，Y 轴不平移
0x00 0x00 0x00 0x00:	Dz=0，Z 轴不平移
0xD4:	CRC16-CCITT 校验低字节
0x0B:	CRC16-CCITT 校验高字节
0x6F, 0xF6:	帧尾

返回数据：

表 26.工具坐标系转换命令的返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x13	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0xC1	CRC16-CCITT 校验低字节
8	0xC2	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

命令#12 获取适配器版本号

此命令用于获取适配器版本号

表 27.获取适配器版本号命令

帧头	有效数据长度	设备地址	保留字节	命令	CRC 低字节	CRC 高字节	帧尾

Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte3	Byte4	Byte5	Byte6	Byte7
0xF6	0x6F	0x03	0x00	0x00	0x14	0x29	0x9E	0x6F	0xF6

返回数据:

表 28. 获取适配器版本号命令的返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x06	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x14	命令字节
6	……	主版本号
7	……	次版本号
8	……	修订版本号
9	……	CRC16-CCITT 校验低字节
10	……	CRC16-CCITT 校验高字节
11	0x6F	帧尾第一字节
12	0xF6	帧尾第二字节

命令#13 初始化传感器

此命令可用来初始化六维力传感器。

表 29. 初始化传感器命令

帧头		有效数据长度	设备地址	保留字节	命令	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte3	Byte4	Byte5	Byte6	Byte7
0xF6	0x6F	0x03	0x00	0x00	0x15	0x08	0x8E	0x6F	0xF6

返回数据:

表 30.初始化传感器命令返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x15	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0x67	CRC16-CCITT 校验低字节
8	0x68	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

命令#14 激活通道 2 传感器

此命令可用来激活挂在通道 2 的六维力传感器

表 31. 激活通道 2 传感器命令

帧头		有效数据长度	设备地址	保留字节	命令	激活或失效	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8
0xF6	0x6F	0x04	0x00	0x00	0x16	0x01	0x34	0x3D	0x6F	0xF6

返回数据:

表 32.激活通道 2 传感器命令数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x16	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0x34	CRC16-CCITT 校验低字节
8	0x3D	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

注：

1. 通道 2 传感器激活后还需要发送初始化命令来初始化所有通道的传感器
2. 激活通道 2 传感器后，通道 2 传感器的采样数据将会被加入到单次采样和连续采样的数据包中

命令#15 获取传感器状态码

此命令可用来获取传感器状态码

表 33. 获取传感器状态码命令

帧头		有效数据长度	设备地址	保留字节	命令	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte3	Byte5	Byte6	Byte7	Byte8
0xF6	0x6F	0x03	0x00	0x00	0x17	0x4A	0xAE	0x6F	0xF6

返回数据:

表 34.获取传感器状态码命令数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	……	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x17	命令字节
6~N	……	4bytes 传感器状态码，若激活通道 2 传感器则会追加 4bytes 通道 2 传感器的状态码

N+1	……	CRC16-CCITT 校验低字节
N+2	……	CRC16-CCITT 校验高字节
N+3	0x6F	帧尾第一字节
N+4	0xF6	帧尾第二字节

表 35. 4 字节状态码解析

Bit number	描述
0	为 1 时表示传感器和 Ethernet 转接板通讯失败，检测不到传感器
1	为 1 时表示获取传感器校正矩阵失败
2	为 1 时表示获取传感器校温补系数失败
3	为 1 时表示获取传感器 ADC 增益失败
4	为 1 时表示传感器输出 ADC 数值错误
5	为 1 时表示传感器零点复位失败
6	为 1 时表示传感器设置 DAC 失败
7	为 1 时表示传感器不存在校正矩阵
8	为 1 是表示传感器输出数据异常
9	为 1 时表示姿态传感器初始化失败
10	为 1 时表示姿态传感器输出数据错误
11	为 1 时表示传感器工作在过载状态
12	为 1 时表示传感器不存在无负载基准电压

命令#16 设置低通滤波参数

此命令可用设置低通滤波参数，共有 0~6 七个滤波等级

表 36. 设置低通滤波参数命令

帧头		有效数据长度	设备地址	保留字节	命令	低通滤波等级	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7	Byte8	Byte9	Byte10
0xF6	0x6F	0x04	0x00	0x00	0x18	.....	.....	.....	0x6F	0xF6

表 37. 低通滤波等级对应的数据输出频率

滤波等级	数据输出速率
0	2000HZ
1	1000HZ
2	500HZ
3	250HZ
4	125HZ
5	62HZ
6	31HZ

以下命令设置低通滤波等级为 6:

F6 6F 04 00 00 18 06 DC 6E 6F F6

解析:

0xF6, 0x6F: 帧头

0x04: 有效数据长度

0x00: 设备地址,0x00 配置通道 1 传感器，0x01 配置通道 2 传感器，0x02 配置两个通道的传感器

0x00: 保留字节

0x18 : 设置低通滤波等级命令

0x06: 低通滤波等级为 6

0xDC: CRC16-CCITT 校验低字节

0x6E: CRC16-CCITT 校验高字节

0x6F, 0xF6: 帧尾

返回数据:

表 38.设置低通滤波参数命令返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x18	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0x1A	CRC16-CCITT 校验低字节
8	0x0E	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

命令#17 设置适配器的 IP 地址

此命令可用设置适配器的 IP 地址

表 39. 设置适配器的 IP 地址命令

帧头		有效数据长度	设备地址	保留字节	命令	IP 地址	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte 6~9	Byte 10	Byte 11	Byte 12	Byte 13
0xF6	0x6F	0x07	0x00	0x00	0x19	.....	.....	.....	0x6F	0xF6

注：修改完 IP 地址后，需要发送保存用户设置命令保存当前的配置到 FLASH 中，然后重新启动适配器

例:

以下命令将适配器 IP 地址修改为 192.168.1.101

F6 6F 07 00 00 19 C0 A8 01 65 43 F5 6F F6

解析:

0xF6, 0x6F: 帧头

0x07: 有效数据长度

0x00: 设备地址

0x00: 保留字节

0x19 : 设置适配器的 IP 地址命令

C0 A8 01 65: 将 IP 修改为 192.168.1.101
0x43: CRC16-CCITT 校验低字节
0xF5: CRC16-CCITT 校验高字节
0x6F, 0xF6: 帧尾
返回数据:

表 40.设置适配器的 IP 地址命令返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x19	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0x0A	CRC16-CCITT 校验低字节
8	0x2D	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

命令#18 设置适配器的掩码

此命令可用设置适配器的掩码

表 41. 设置适配器的掩码命令

帧头		有效数据长度	设备地址	保留字节	命令	掩码	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte 6~9	Byte 10	Byte 11	Byte 12	Byte 13
0xF6	0x6F	0x07	0x00	0x00	0x1A	.....	.....	.....	0x6F	0xF6

注：修改完掩码后，需要发送保存用户设置命令保存当前的配置到 FLASH 中，然后重新启动适配器

例:
以下命令将适配器掩码修改为 255.255.255.0
F6 6F 07 00 00 1A FF FF FF 00 05 34 6F F6
解析:
0xF6, 0x6F: 帧头
0x07: 有效数据长度
0x00: 设备地址
0x00: 保留字节
0x1A : 设置适配器掩码命令
FF FF FF 00: 将掩码修改为 255.255.255.0
0x05: CRC16-CCITT 校验低字节
0x34: CRC16-CCITT 校验高字节
0x6F, 0xF6: 帧尾

返回数据:

表 42.设置适配器的掩码命令返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x1A	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0x59	CRC16-CCITT 校验低字节
8	0x78	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

命令#20 设置适配器的网关

此命令可用设置适配器的网关

表 43. 设置适配器的网关命令

帧头		有效数据长度	设备地址	保留字节	命令	网关	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte 6~9	Byte 10	Byte 11	Byte 12	Byte 13
0xF6	0x6F	0x07	0x00	0x00	0x1B	.....	.....	.....	0x6F	0xF6

注：修改完网关后，需要发送保存用户设置命令保存当前的配置到 FLASH 中，然后重新启动适配器

例:

以下命令将适配器掩码修改为 192.168.1.1

F6 6F 07 00 00 1B C0 A8 01 01 E2 9D 6F F6

解析:

0xF6, 0x6F: 帧头

0x07: 有效数据长度

0x00: 设备地址

0x00: 保留字节

0x1B : 设置适配器网关命令

C0 A8 01 01: 将网关修改为 192.168.1.1

0xE2: CRC16-CCITT 校验低字节

0x9D: CRC16-CCITT 校验高字节

0x6F, 0xF6: 帧尾

返回数据:

表 44.设置适配器的网关命令返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度

3	0x00	设备地址
4	0x00	保留字节
5	0x1B	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0x68	CRC16-CCITT 校验低字节
8	0x4B	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

命令#21 设置适配器的端口号

此命令可用设置适配器的端口号

表 45. 设置适配器的端口号命令

帧头		有效数据长度	设备地址	保留字节	命令	端口号	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte 6~7	Byte 8	Byte 9	Byte 10	Byte 11
0xF6	0x6F	0x05	0x00	0x00	0x1C	.....	.....	.....	0x6F	0xF6

注：修改完端口号后，需要发送保存用户设置命令保存当前的配置到 FLASH 中，然后重新启动适配器

例:

以下命令将适配器端口修改为 8080

F6 6F 05 00 00 1C 90 1F 3B DC 6F F6

解析:

0xF6, 0x6F: 帧头

0x05: 有效数据长度

0x00: 设备地址

0x00: 保留字节

0x1C : 设置适配器端口命令

90 1F: 将端口修改为 8080

0x3B : CRC16-CCITT 校验低字节

0xDC : CRC16-CCITT 校验高字节

0x6F, 0xF6: 帧尾

返回数据:

表 46.设置适配器的网关命令返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x1C	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0xFF	CRC16-CCITT 校验低字节

8	0xD2	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

命令#22 设置适配器解析模式

此命令可用于将适配器的解析模式切换为 ASCII 模式

表 47. 设置适配器解析的模式

帧头		有效数据长度	设备地址	保留字节	命令	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte3	Byte4	Byte5	Byte6	Byte7
0xF6	0x6F	0x03	0x00	0x00	0x1E	0x63	0x3F	0x6F	0xF6

返回数据:

表 49.设置适配器的网关命令返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x1E	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0x9D	CRC16-CCITT 校验低字节
8	0xB4	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

命令#22 设置中值滤波深度

此命令可用设置中值滤波深度

表 48. 设置中值滤波深度命令

帧头		有效数据长度	设备地址	保留字节	命令	中值滤波深度 (0~64)	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte 7	Byte 8	Byte 9	Byte 10
0xF6	0x6F	0x04	0x00	0x00	0x20	.....	.....	.....	0x6F	0xF6

注：一共有 0~64 个滤波深度，0 为关闭中值滤波。中值滤波能有效抑制噪声消除孤立的跳变点，使曲线变化更平滑但会降低瞬时响应速率

例:

以下命令将中值滤波深度设置为 10

F6 6F 04 00 00 20 0A 6C 23 6F F6

解析:

0xF6, 0x6F: 帧头

0x04: 有效数据长度
0x00: 设备地址
0x00: 保留字节
0x20 : 设置中值滤波深度命令
0x0A: 将中值滤波深度设置为 10
0x6C : CRC16-CCITT 校验低字节
0x23 : CRC16-CCITT 校验高字节
0x6F, 0xF6: 帧尾

返回数据:

表 49.设置中值滤波深度命令返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x20	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0x07	CRC16-CCITT 校验低字节
8	0x92	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

命令#23 清除传感器 IO 报警信号

此命令可清除传感器 IO 报警信号，传感器检测到力和力矩的数值超过用户设定的阈值时，端口 12 和 13 便会导通，在触发 IO 报警信号后用户需要使用该命令清除报警信号，然后传感器进行下一次的报警监控。

表 50. 清除传感器的 IO 报警信号

帧头		有效数据长度	设备地址	保留字节	命令	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte 6	Byte 7	Byte 8	Byte 9
0xF6	0x6F	0x03	0x00	0x00	0x23	0xD8	0x9D	0x6F	0xF6

返回数据:

表 51.清除传感器 IO 报警信号命令返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x23	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0x54	CRC16-CCITT 校验低字节

8	0xC7	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

命令#24 设置 IO 报警的阈值

此命令可设置传感器 IO 报警的阈值。

表 52. 设置 IO 报警预值

帧头		有效数据长度	设备地址	保留字节	命令	6 个轴的报警阈值	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte6~29	Byte 30	Byte 31	Byte 32	Byte 33
0xF6	0x6F	0x1B	0x00	0x00	0x24	.....	.....	.....	0x6F	0xF6

报警阈值大小为 6*4bytes，每个轴 4bytes，分别对应 Fx、Fy、Fz、Mx、My、Mz 的报警阈值，且该值等于真实的值*1000。

在 IO 报警信号触发后，需要发送清除 IO 报警命令使传感器恢复监控功能

例:

以下命令将 X、Y、Z 轴的力和扭矩的阈值分别设置为 200N、200N、200N、10N*m、10N*m、10N*m。

F6 6F 1B 00 00 24 40 0D 03 00 40 0D 03 00 40 0D 03 00 10 27 00 00 10 27 00 00 10 27 00 00 D3 59 6F F6

解析:

0xF6, 0x6F: 帧头

0x1B: 有效数据长度

0x00: 设备地址

0x00: 保留字节

0x24 : 清除传感器的 IO 报警命令

0x40 0x0D 0x03 0x00: 200000，对应 Fx 的报警阈值为 200N

0x40 0x0D 0x03 0x00: 200000，对应 Fy 的报警阈值为 200N

0x40 0x0D 0x03 0x00: 200000，对应 Fz 的报警阈值为 200N

0x10 0x27 0x00 0x00: 10000，对应 Mx 的报警阈值为 10N*m

0x10 0x27 0x00 0x00: 10000，对应 My 的报警阈值为 10N*m

0x10 0x27 0x00 0x00: 10000，对应 Mz 的报警阈值为 10N*m

0xD3 : CRC16-CCITT 校验低字节

0x59: CRC16-CCITT 校验高字节

0x6F, 0xF6: 帧尾

返回数据:

表 53.设置中值滤波深度命令返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x24	命令字节

6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0xC3	CRC16-CCITT 校验低字节
8	0x5E	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

命令#25 使能传感器 IO 报警

此命令可使能或关闭传感器 IO 报警功能

表 54. 使能传感器 IO 报警

帧头		有效数据长度	设备地址	保留字节	命令	使能或关闭	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte 7	Byte 8	Byte 9	Byte 10
0xF6	0x6F	0x04	0x00	0x00	0x25	0x01/ 0x00	.....	.....	0x6F	0xF6

默认 IO 报警功能并不使能，使用前需要发送该命令使能 IO 报警功能。在 IO 报警信号触发后，需要发送清除 IO 报警命令使传感器恢复监控功能。

例:

以下命令用于使能传感器 IO 报警功能:

F6 6F 04 00 00 25 01 F2 6D 6F F6

解析:

0xF6, 0x6F: 帧头

0x04: 有效数据长度

0x00: 设备地址

0x00: 保留字节

0x25 : 使能/关闭传感器 IO 报警命令

0x01: 使能报警

0xF2 : CRC16-CCITT 校验低字节

0x6D: CRC16-CCITT 校验高字节

0x6F, 0xF6: 帧尾

表 55.使能传感器 IO 报警命令返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x25	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0xF2	CRC16-CCITT 校验低字节
8	0x6D	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节

10	0xF6	帧尾第二字节

命令#26 获取触发 IO 报警的轴信息

此命令可获取触发 IO 报警的轴的信息。

表 56. 获取触发 IO 报警的轴的信息

帧头		有效数据长度	设备地址	保留字节	命令	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte 6	Byte 7	Byte 8	Byte 9
0xF6	0x6F	0x03	0x00	0x00	0x26	0x38	0x88	0x6F	0xF6

在传感器触发 IO 报警后，用户可使用该命令获知是哪个轴超过了用户设置的阈值。

返回数据:

表 57.获取触发 IO 报警的轴的信息命令返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x0A	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x26	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7~12	.....	6bytes 对应 6 个轴的标志位，0x01 表示超过阈值
13	0xC3	CRC16-CCITT 校验低字节
14	0x5E	CRC16-CCITT 校验高字节
15	0x6F	帧尾第一字节
16	0xF6	帧尾第二字节

命令#27 手动触发传感器 IO 报警

此命令可手动触发 IO 报警。

表 58.手动触发传感器 IO 报警

帧头		有效数据长度	设备地址	保留字节	命令	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte 6	Byte 7	Byte 8	Byte 9
0xF6	0x6F	0x03	0x00	0x00	0x28	0xF6	0x69	0x6F	0xF6

返回数据:

表 59.手动触发传感器 IO 报警返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址

4	0x00	保留字节
5	0x28	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0xAE	CRC16-CCITT 校验低字节
8	0x1B	CRC16-CCITT 校验高字节
9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

命令#28 设置 IO 报警端口为常开或常闭合模式

此命令用于设置 IO 端口的模式。

表 60. 设置 IO 报警端口为常开或常闭合模式

帧头		有效数据长度	设备地址	保留字节	命令	常开或常闭	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte 7	Byte 8	Byte 9	Byte 10
0xF6	0x6F	0x03	0x00	0x00	0x27	0x01/ 0x00	.....	.....	0x6F	0xF6

默认为常开模式，即报警信号没有触发时，OUT 端口和 COM 端口是断开的。设置完成后可以发送保存用户设置命令将配置保存到 FLASH 中以后上电都使用该配置。

例:

以下命将 IO 端口配置为常闭模式。

F6 6F 04 00 00 27 00 B1 1B 6F F6

解析:

0xF6, 0x6F: 帧头

0x04: 有效数据长度

0x00: 设备地址

0x00: 保留字节

0x27 : 设置中值滤波深度命令

0x00: 将将 IO 端口配置为常闭模式。

0xB1 : CRC16-CCITT 校验低字节

0x1B : CRC16-CCITT 校验高字节

0x6F, 0xF6: 帧尾

返回数据:

表 61.设置 IO 端口的模式返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x04	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0x27	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7	0x90	CRC16-CCITT 校验低字节
8	0x0B	CRC16-CCITT 校验高字节

9	0x6F	帧尾第一字节
10	0xF6	帧尾第二字节

命令#29 获取过载次数信息

此命令可获取过载次数信息。

表 62. 获取过载次数信息

帧头		有效数据长度	设备地址	保留字节	命令	命令 2	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte 7	Byte 8	Byte 9	Byte 10
0xF6	0x6F	0x04	0x00	0x00	0xD4	0xA6	0x0F	0x88	0x6F	0xF6

用户可使用该命令获知是各个轴的过载次数信息,每 20ms 检测一次。最大记录连续过载时间 994 天.

返回数据:

表 63.获取过载次数信息命令返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节
2	0x1C	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0xD4	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7~30	.....	6*4bytes 对应 6 个轴的过载次数
31	0xC3	CRC16-CCITT 校验低字节
32	0x5E	CRC16-CCITT 校验高字节
33	0x6F	帧尾第一字节
34	0xF6	帧尾第二字节

命令#30 获取过载峰值信息

此命令可获取过载峰值信息。

表 62. 获取过载峰值信息

帧头		有效数据长度	设备地址	保留字节	命令	命令 2	CRC 低字节	CRC 高字节	帧尾	
Byte 0	Byte 1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte 7	Byte 8	Byte 9	Byte 10
0xF6	0x6F	0x04	0x00	0x00	0xD8	0xA6	0x62	0xCD	0x6F	0xF6

用户可使用该命令获知是各个轴的过载峰值信息,每 20ms 检测一次。

返回数据:

表 63.获取过载峰值信息命令返回数据

字节号	数据	描述
0	0xF6	帧头第一字节
1	0x6F	帧头第二字节

2	0x1C	有效数据长度
3	0x00	设备地址
4	0x00	保留字节
5	0xD8	命令字节
6	0x01	应答字节，成功为 0x01，失败为 0x00
7~30	.....	6*4bytes 对应 6 个轴的过载次数
31	0xC3	CRC16-CCITT 校验低字节
32	0x5E	CRC16-CCITT 校验高字节
33	0x6F	帧尾第一字节
34	0xF6	帧尾第二字节

2.2.3 ASCII 数据格式

该模式下命令格式为 “”<Cmd arg0 arg1 ... argX>“”(不包括双引号), Ethernet 适配器出厂默认为二进制模式，可以发送命令 F6 6F 03 00 00 1E 63 3F 6F F6 切换到 ASCII 解析模式，然后发送命令<SAVE>保存当前模式到 FLASH 中，下次上电后则为 ASCII 解析模式。

2.2.4 命令列表

表 62. ASCII 命令列表

命令	命令字符串	描述
零点校正	ZERO	执行传感器零点校正
单次测量	SINGLE	单次输出测量数据
连续测量	CONTINUE	连续输出测量数据
停止连续测量	STOP	停止连续测量
保存用户设置	SAVE	保存用户设置
切换解析模式	BINARY	切换到二进制解析模式
获取固件版本号	VERSION	获取 Ethernet 适配器和传感器的固件版本号
获取传感器序列号	SERIAL_NUMBER	获取传感器的序列号
设置低通滤波等级	FILTER	设置低通滤波等级
打开或关闭卡尔曼滤波	KALMAN	打开或关闭卡尔曼滤波
设置卡尔曼滤波参数	KALMAN_PARA	设置卡尔曼滤波参数
工具坐标转换	TOOL	传感器工具坐标转换
激活端口 2 传感器	ACTIVE	该命令用于激活通道 2 的传感器，默认只激活通道 1；当 Ethernet 适配器挂载两个六维力传感器时需要使用该命令激活通道 2 的传感器
初始化传感器	INIT	该命令用于枚举和初始化传感器，在激活通道 2 的传感器后需要发送该命令初始化通道 2 传感器
设置过程数据的格式	FORMAT	设置过程数据的格式
设置输出过程数据的顺序	ORDER	设置输出过程数据的顺序，默认为 Fx->Fy->Fz->Mx->My->Mz
设置返回字符串的结束符	EOF	设置返回字符串的结束符，默认为\r\n，ASCII 码为 13 10
获取传感器状态	ERROR_CODE	获取传感器状态码
设置适配器的 IP	IP	设置适配器的 IP
设置适配器的掩码	MASK	设置适配器的掩码

设置适配器的网关	GATEWAY	设置适配器的网关
设置适配器的端口号	PORT	设置适配器的端口号
清除 IO 报警信号	ALARM_CLEAR	清除传感器的 IO 报警信号
设置 IO 报警信号的阈值	ALARM_THRE_VALUE	设置 IO 报警信号的阈值
使能传感器 IO 报警	ALARM_ON	使能传感器 IO 报警，出厂默认为不使能
获取触发 IO 报警的轴信息	ALARM_AXIS	获取触发 IO 报警的轴,0~5 分别代表 Fx、Fy、Fz、Mx、My、Mz
设置 IO 报警端口为常开或常闭合模式	ALARM_NORMAL_OPEN	默认为常开模式，即报警信号没有触发时，OUT 端口和 COM 端口是断开的
手动触发传感器 IO 报警	ALARM_SET	该命令用于手动触发 IO 报警
获取过载次数信息	OVERLOAD_COUNT	获取过载次数信息,0~5 分别代表 Fx、Fy、Fz、Mx、My、Mz
获取过载峰值信息	OVERLOAD_PEAK	获取过载峰值信息,0~5 分别代表 Fx、Fy、Fz、Mx、My、Mz

命令#1 零点校正

命令格式:<ZERO>或<ZERO 设备地址>

返回数据:成功返回 1 失败返回 0

说明:

此命令用来复位传感器的零点输出，传感器安装完毕后发送该命令即可复位传感器的零点。若 Ethernet 适配器挂载两个六维力传感器，则<ZERO>用于零点复位两个通道的传感器；<ZERO 0>用于复位通道 1 的传感器；<ZERO 1>用于复位通道 2 的传感器。

命令#2 单次测量

命令格式:<SINGLE>

说明:

此命令将启动传感器进行一次单次测量。

命令#3 连续测量

命令格式:<CONTINUE>

说明:

此命令将启动传感器进行连续测量。默认出厂单次/连续测量命令在 Ethernet 适配器挂载单个六维力传感器时数据格式为:Fx,Fy,Fz,Mx,My,Mz；可以通过命令<FORMAT>设置返回过程数据的格式。在适配器挂载两个传感器时数据格式为:Fx,Fy,Fz,Mx,My,Mz,SampleCount,Fx2,Fy2,Fz2,Mx2,My2,Mz2,SampleCount2，Ethernet 转接板每次接收到传感器一帧数据会加 1，通过 4 个十进制位的 SampleCount 用户可以根据该数值判断当前接收到的数据和上一次接收到的数据是否是同一帧。

命令#4 停止连续测量

命令格式:<STOP>

返回数据:成功返回 1 失败返回 0

说明:

此命令将停止传感器的连续测量输出。

命令#5 保存用户设置

命令格式:<SAVE>

返回数据:成功返回 1 失败返回 0

说明:

此命令可以将用户设置保存到传感器内部的 Flash 中，保存的用户设置断电不会丢失，并将在每一次传感器上电时被自动加载。

命令#6 切换命令解析模式

命令格式:<BINARY>

返回数据:成功返回 1 失败返回 0

说明:

此命令用于将适配器的解析模式切换到二进制解析模式,切换为二进制模式后需要按二进制模式下的命令格式发送命令和传感器通讯。

命令#7 获取固件版本号

命令格式:<VERSION>

返回数据:返回 Ethernet 适配器和传感器的固件版本号

说明:

此命令用于获取 Ethernet 适配器和传感器的固件版本号。

命令#8 获取传感器序列号

命令格式:<SERIAL_NUMBER>

返回数据:返回传感器的序列号

说明:

此命令用于获取传感器的序列号。

命令#9 设置低通滤波等级

命令格式:<FILTER 设备地址 低通滤波等级>

返回数据:成功返回 1 失败返回 0

说明:

此命令可用设置低通滤波参数，一共有 0~6，七个等级，参考表 39. 低通滤波等级对应的数据输出频率。设备地址用于指定设置哪个通道的传感器，2 为同时设置通道 1 和通道 2。例：<FILTER 0 6> 设置通道 1 的传感器的低通滤波等级为 6。

命令#10 打开或关闭卡尔曼滤波

命令格式:<KALMAN 设备地址 0 或 1>

返回数据:成功返回 1 失败返回 0

说明:

此命令用于打开或关闭卡尔曼滤波，设备地址用于指定设置哪个通道的传感器，2 为同时设置通道 1 和通道 2。例：<KALMAN 0 1>打开通道 1 传感器的卡尔曼滤波;<KALMAN 0 0>关闭通道 1 传感器的卡尔曼滤波。

命令#11 设置卡尔曼滤波参数

命令格式:<KALMAN_PARA 设备地址 力数据的卡尔曼滤波参数 力矩数据的卡尔曼滤波参数>

返回数据:成功返回 1 失败返回 0

说明:

此命令用于设置力和力矩的卡尔曼滤波参数，设备地址用于指定设置哪个通道的传感器，0 为设置通道 1 的传感器，2 为同时设置通道 1 和通道 2。力数据和力矩数据的卡尔曼滤波参数如表 51、表 52。

表 63.力数据的卡尔曼滤波参数

名称	功能	描述
kalman_K	力的新数据的权重	力的新数据的权重，可配置为 1 ~ 100 默认值：20
kalman_threshold	力的噪声数据的阈值	力的噪声数据的阈值，单位 N 默认值：0.25
num_check	连续超过噪声阈值的力的帧数	连续超过噪声阈值的力的帧数 默认值：3

表 52.力矩数据的卡尔曼滤波参数

kalman_K	力矩新数据的权重	力矩新数据的权重，可配置为 1 ~ 100 默认值：20
kalman_threshold	力矩噪声数据的阈值	力矩噪声数据的阈值，单位 N*m 默认值：0.002
num_check	连续超过噪声阈值的力矩的帧数	连续超过噪声阈值的力矩的帧数 默认值：3

注： a. 新数据的权重：适当降低该数值可降低数据噪声，但会降低瞬间响应速度。

b. 噪声数据的阈值：该值应该略大于当前传感器的背景噪声值。

c. 连续超过噪声阈值的的帧数：数据值大于噪声数据的阈值的连续帧数量超过该数值时数据有效，适当提高该值可以减小数据的跳动，但会损失一些瞬时响应速度。

例：以下命令将通道 1 的传感器的力的卡尔曼滤波参数设置为：kalman_K = 20, kalman_threshold = 0.8, num_check=2，力矩的卡尔曼滤波参数设置为：kalman_K = 20, kalman_threshold = 0.002, num_check=2。

<KALMAN_PARA 0 20,0.8,2 20,0.002,2>

命令#12 工具坐标系转换

命令格式:<TOOL 设备地址 工具坐标系转换参数>

返回数据:成功返回 1 失败返回 0

说明:

此命令可用于工具坐标系转换，坐标系围绕某个轴旋转时，顺时针为正，旋转顺序为 X 轴->Y 轴->Z 轴。工具坐标系转换参数格式为:X 轴平移距离,Y 轴的平移距离,Z 轴的平移距离,X 轴的旋转角度,Y 轴的旋转角度,Z 轴的旋转角度。

例：以下命令将通道 1 的传感器的工具坐标系绕 Z 轴顺时针旋转 90 度并将工具坐标系向 X 轴正方向平移 0.02 米。

<TOOL 0 0.02,0,0,0,0,90>

命令#13 激活通道 2 传感器

命令格式:<ACTIVE 0 或 1>

返回数据:成功返回 1 失败返回 0

说明:

若 Ethernet 适配器挂载两个六维力传感器，则需要发送激活命令激活通道 2 的传感器，然后发送<INIT>命令初始化通道 2 传感器。例:以下命令将通道 2 的传感器激活 <ACTIVE 1>

命令#14 初始化传感器

命令格式:<INIT>

返回数据:成功返回 1 失败返回 0

说明:

若 Ethernet 适配器挂载两个六维力传感器，则需要发送激活命令激活通道 2 的传感器，然后发送<INIT>命令初始化通道 2 传感器。

命令#15 设置过程数据的格式

命令格式:<FORMAT 格式化字符串>

返回数据:成功返回 1 失败返回 0

说明:

该命令用于设置 Ethernet 适配器返回的力/力矩数据的格式,格式化字符串符合 C 语言格式化字符串规则。出厂默认的数据格式为"%.3f,%.3f,%.3f,%.3f,%.3f,%.3f"，在挂载两个传感器时数据格式为"%.3f,%.3f,%.3f,%.3f,%.3f,%.3f,%d,%.3f,%.3f,%.3f,%.3f,%.3f,%.3f,%d"，%d 对应每个传感器的采样计数。用户可以通过该命令设置输出数据的格式，使之能够匹配机器人内部变量的格式。

例 1：在挂载单个传感时发送命令<FORMAT {X %.3f,Y %.3f,Z %.3f,A %.3f,B %.3f,C %.3f}>则输出的力和力矩数据格式为:{X -1.081,Y -0.029,Z -19.760,A 0.072,B -0.167,C 0.004}。

例 2：通过命令<FORMAT %.3f,%.3f,%.3f>则适配器只输出 X、Y、Z 轴力的数据。

例 3：通过命令<FORMAT %.1f,%.1f,%.1f,%.1f,%.1f,%.1f>则适配器输出力和力矩的数据只保留小数点后一位。

命令#16 设置输出过程数据的顺序

命令格式:<ORDER XYZ 轴力和力矩的编号>

返回数据:成功返回 1 失败返回 0

说明:

该命令用于设置输出过程数据的顺序,出厂默认为 Fx->Fy->Fz->Mx->My->Mz 即 XYZ 轴力和力矩的编号为 0,1,2,3,4,5；

在挂载两个六维力时顺序为 Fx1->Fy1->Fz1->Mx1->My1->Mz1->Fx2->Fy2->Fz2->Mx2->My2->Mz2 即 XYZ 轴力和力矩的编号为 0,1,2,3,4,5,6,7,8,9,10,11；

例：在挂载单个六维力传感器时以下命令将力和力矩的输出顺序设置为 Fz->Fx->Fy->Mz->Mx->My <ORDER 2,0,1,5,3,4>

命令#17 设置返回字符串的结束符

命令格式:<EOF 结束符的 ASCII 码>

返回数据:成功返回 1 失败返回 0

说明:

该命令用于设置 Ethernet 设配器返回的字符串的结束符,出厂默认为\r\n 即 ASCII 码为 13 10。

例:以下命令将适配器所有返回字符串的结束符设置为\r\n <EOF 13,10>

命令#18 获取传感器状态

命令格式:<ERROR_CODE>

返回数据:返回适配器的状态码，详细参考表 37

说明:

该命令用于获取适配器的状态码

命令#19 设置适配器的 IP

命令格式:<IP IP 地址>

返回数据:成功返回 1 失败返回 0

说明:

该命令用于设置适配器的 IP,设置网络参数后需要发送<SAVE>命令将通讯参数保存到 FLASH 中, 重新上电后参数生效。例:以下命令将适配器的 IP 地址设置为 192.168.0.100 <IP 192.168.0.100>

命令#20 设置适配器的子网掩码

命令格式:<MASK 地址>

返回数据:成功返回 1 失败返回 0

说明:

该命令用于设置适配器的子网掩码,设置网络参数后需要发送<SAVE>命令将通讯参数保存到 FLASH 中, 重新上电后参数生效。例:以下命令将适配器的 MASK 地址设置为 255.255.255.0 <MASK 255.255.255.0>

命令#21 设置适配器的网关

命令格式:<GATEWAY 地址>

返回数据:成功返回 1 失败返回 0

说明:

该命令用于设置适配器的网关,设置网络参数后需要发送<SAVE>命令将通讯参数保存到 FLASH 中, 重新上电后参数生效。例:以下命令将适配器的 GATEWAY 地址设置为 192.168.0.1 <GATEWAY 192.168.0.1>

命令#22 设置适配器的端口号

命令格式:<PORT 端口号>

返回数据:成功返回 1 失败返回 0

说明:

该命令用于设置适配器的端口号,设置网络参数后需要发送<SAVE>命令将通讯参数保存到 FLASH 中, 重新上电后参数生效。例:以下命令将适配器的 PORT 设置为 12345 <PORT 12345>

命令#23 设置中值滤波深度

命令格式:<MEDIAN_FILTER 设备地址 中值滤波深度>

返回数据:成功返回 1 失败返回 0

说明:

一共有 0~64 个滤波深度，0 为关闭中值滤波。中值滤波能有效抑制噪声消除孤立的跳变点，使曲线变化更平滑但会降低瞬时响应速率

命令#24 清除 IO 报警信号

命令格式:<ALARM_CLEAR 设备地址>

返回数据:成功返回 1 失败返回 0

说明:

在 IO 报警信号触发后，需要发送清除 IO 报警命令使传感器恢复监控功能

命令#25 设置 IO 报警信号的阈值

命令格式:<ALARM_THRE_VALUE 设备地址 Fx,Fy,Fz,Mx,My,Mz>

返回数据:成功返回 1 失败返回 0

说明:

在 IO 报警信号触发后，需要发送清除 IO 报警命令使传感器恢复监控功能

例：将通道 1 的传感器的 X、Y、Z 轴的力和扭矩的报警阈值分别设置为 200N、200N、200N、10N*m、10N*m、10N*m。

<ALARM_THRE_VALUE 0 200,200,200,10,10,10>

命令#26 使能传感器 IO 报警

命令格式:<ALARM_ON 设备地址 0 或 1>

返回数据:成功返回 1 失败返回 0

说明:

默认 IO 报警功能并不使能，使用前需要发送该命令使能 IO 报警功能。IO 端口默认为常开模式，即在没有触发报警时，OUT 端口和 COM 端口是断开的，用户可以使用 ALARM_NORMAL_OPEN 配置 IO 端口的模式。在 IO 报警信号触发后，需要发送清除 IO 报警命令使传感器恢复监控功能。

例：使能通道 1 的传感器的 IO 报警功能

<ALARM_ON 0 1>

命令#27 获取触发 IO 报警的轴信息

命令格式:<ALARM_AXIS 设备地址>

返回数据:返回 6 个轴的报警信息，例：1,0,0,0,0,0 表示 X 轴的力超过了用户设置的阈值

说明:

在传感器触发 IO 报警后，用户可使用该命令获知是哪个轴超过了用户设置的阈值。

命令#28 手动触发传感器 IO 报警

命令格式:<ALARM_SET>

返回数据:成功返回 1 失败返回 0

说明:

用户可以通过该命令手动触发 IO 信号，通过 ALARM_CLEAR 命令将信号清除

命令#29 设置 IO 报警端口为常开或常闭合模式

命令格式:<ALARM_NORMAL_OPEN 1 或 0>

返回数据:成功返回 1 失败返回 0

说明:

用户可以通过该命令设置 IO 端口为常开或常闭，设置完成后可以发送<SAVE>命令将该配置保存到 FLASH 中，以后传感器上电都使用该配置

例：以下命令将 IO 端口设置为常开模式，即在没有报警信号时，OUT 端和 COM 端是断开的

<ALARM_NORMAL_OPEN 1>

命令#30 获取过载次数信息

命令格式:<OVERLOAD_COUNT 设备地址>

返回数据:返回 6 个轴的过载次数信息，例：1,0,0,0,0,0 表示 X 轴的力超过了用户设置的阈值一个检测周期

说明:

在传感器超过默认量程或设定的过载阈值时，过载计数器会每 20ms 检测一次并累加。最大记录连续过载时间 994 天.

命令#31 获取过载峰值信息

命令格式:<OVERLOAD_PEAK 设备地址>

返回数据:返回 6 个轴的过载峰值信息，例：666,0,0,0,0,0 表示 X 轴的力超过了默认量程或设定的过载阈值的最大峰值.

说明:

在传感器超过默认量程或设定的过载阈值时，过载计数器会每 20ms 检测一次并累加。

修订历史记录

表 64. 规格书修订历史记录

Date	Revision	Description
2019/09/20	2.0	将命令说明和规格说明分成两个手册
2021/09/29	2.1	添加获取过载计数和峰值指令

附录

CRC16-CCITT 的 C 语言实现

实现 1:

```
#################################################################
#include<stdio.h>
/**
Flash Space: Small
Calculation Speed: Slow
*/
/*Function Name:          crc_cal_by_bit          //按位计算CRC
  Function Parameters:    unsigned char* ptr      //数据缓冲区指针
                          unsigned char len       //数据长度
  Return Value:           unsigned int
  Polynomial:             CRC-CCITT 0x1021
*/
unsigned int crc_cal_by_bit(unsigned char* ptr, unsigned char len)
{
    #define CRC_CCITT  0x1021
    unsigned int crc = 0xffff;

    while(len-- != 0)
    {
        for(unsigned char i = 0x80; i != 0; i /= 2)
        {
            crc *= 2;
            if((crc&0x10000) !=0)
                crc ^= 0x11021;
            if((*ptr&i) != 0)
                crc ^= CRC_CCITT;
        }
        ptr++;
    }
    return crc;
}
#################################################################
```

实现 2:

```
################################################################
#include<stdio.h>
/**
Flash Space: Medium
Calculation Speed: Medium
*/
/* Function Name:         crc_cal_by_halfbyte    //按半字节计算CRC
   Function Parameters:   unsigned char* ptr     //数据缓冲区指针
                          unsigned char len      //数据长度
   Return Value:          unsigned int
   Polynomial:            CRC-CCITT  0x1021
*/
unsigned int crc_cal_by_halfbyte(unsigned char* ptr, unsigned char len)
{
    unsigned short crc = 0xffff;

    while(len-- != 0)
    {
        unsigned char high = (unsigned char)(crc/4096);
        crc <<= 4;
        crc ^= crc_ta_4[high^(*ptr/16)];
        high = (unsigned char)(crc/4096);
        crc <<= 4;
        crc ^= crc_ta_4[high^(*ptr&0x0f)];
        ptr++;
    }
    return crc;
}

unsigned int crc_ta_4[16]={ /* CRC半字节表 */
    0x0000,0x1021,0x2042,0x3063,0x4084,0x50a5,0x60c6,0x70e7,
    0x8108,0x9129,0xa14a,0xb16b,0xc18c,0xd1ad,0xe1ce,0xf1ef,
};
################################################################
```

实现 3:

```
#####################################################################
#include<stdio.h>
/**
Flash Space: Large
Calculation Speed: Fast
*/
/* Function Name:          crc_cal_by_byte        //按字节计算CRC
   Function Parameters:    unsigned char* ptr     //数据缓冲区指针
                           unsigned char len      //数据长度
   Return Value:           unsigned int
   Polynomial:             CRC-CCITT 0x1021
*/
unsigned int crc_ta_8[256]={ /* CRC字节表 */
    0x0000, 0x1021, 0x2042, 0x3063, 0x4084, 0x50a5, 0x60c6, 0x70e7,
    0x8108, 0x9129, 0xa14a, 0xb16b, 0xc18c, 0xd1ad, 0xe1ce, 0xf1ef,
    0x1231, 0x0210, 0x3273, 0x2252, 0x52b5, 0x4294, 0x72f7, 0x62d6,
    0x9339, 0x8318, 0xb37b, 0xa35a, 0xd3bd, 0xc39c, 0xf3ff, 0xe3de,
    0x2462, 0x3443, 0x0420, 0x1401, 0x64e6, 0x74c7, 0x44a4, 0x5485,
    0xa56a, 0xb54b, 0x8528, 0x9509, 0xe5ee, 0xf5cf, 0xc5ac, 0xd58d,
    0x3653, 0x2672, 0x1611, 0x0630, 0x76d7, 0x66f6, 0x5695, 0x46b4,
    0xb75b, 0xa77a, 0x9719, 0x8738, 0xf7df, 0xe7fe, 0xd79d, 0xc7bc,
    0x48c4, 0x58e5, 0x6886, 0x78a7, 0x0840, 0x1861, 0x2802, 0x3823,
    0xc9cc, 0xd9ed, 0xe98e, 0xf9af, 0x8948, 0x9969, 0xa90a, 0xb92b,
    0x5af5, 0x4ad4, 0x7ab7, 0x6a96, 0x1a71, 0x0a50, 0x3a33, 0x2a12,
    0xdbfd, 0xcbdc, 0xfbbf, 0xeb9e, 0x9b79, 0x8b58, 0xbb3b, 0xab1a,
    0x6ca6, 0x7c87, 0x4ce4, 0x5cc5, 0x2c22, 0x3c03, 0x0c60, 0x1c41,
    0xedae, 0xfd8f, 0xcdec, 0xddcd, 0xad2a, 0xbd0b, 0x8d68, 0x9d49,
    0x7e97, 0x6eb6, 0x5ed5, 0x4ef4, 0x3e13, 0x2e32, 0x1e51, 0x0e70,
    0xff9f, 0xefbe, 0xdfdd, 0xcffc, 0xbf1b, 0xaf3a, 0x9f59, 0x8f78,
    0x9188, 0x81a9, 0xb1ca, 0xa1eb, 0xd10c, 0xc12d, 0xf14e, 0xe16f,
    0x1080, 0x00a1, 0x30c2, 0x20e3, 0x5004, 0x4025, 0x7046, 0x6067,
    0x83b9, 0x9398, 0xa3fb, 0xb3da, 0xc33d, 0xd31c, 0xe37f, 0xf35e,
    0x02b1, 0x1290, 0x22f3, 0x32d2, 0x4235, 0x5214, 0x6277, 0x7256,
    0xb5ea, 0xa5cb, 0x95a8, 0x8589, 0xf56e, 0xe54f, 0xd52c, 0xc50d,
    0x34e2, 0x24c3, 0x14a0, 0x0481, 0x7466, 0x6447, 0x5424, 0x4405,
    0xa7db, 0xb7fa, 0x8799, 0x97b8, 0xe75f, 0xf77e, 0xc71d, 0xd73c,
    0x26d3, 0x36f2, 0x0691, 0x16b0, 0x6657, 0x7676, 0x4615, 0x5634,
    0xd94c, 0xc96d, 0xf90e, 0xe92f, 0x99c8, 0x89e9, 0xb98a, 0xa9ab,
    0x5844, 0x4865, 0x7806, 0x6827, 0x18c0, 0x08e1, 0x3882, 0x28a3,
    0xcb7d, 0xdb5c, 0xeb3f, 0xfb1e, 0x8bf9, 0x9bd8, 0xabbb, 0xbb9a,
    0x4a75, 0x5a54, 0x6a37, 0x7a16, 0x0af1, 0x1ad0, 0x2ab3, 0x3a92,
    0xfd2e, 0xed0f, 0xdd6c, 0xcd4d, 0xbdaa, 0xad8b, 0x9de8, 0x8dc9,
    0x7c26, 0x6c07, 0x5c64, 0x4c45, 0x3ca2, 0x2c83, 0x1ce0, 0x0cc1,
    0xef1f, 0xff3e, 0xcf5d, 0xdf7c, 0xaf9b, 0xbfba, 0x8fd9, 0x9ff8,
```

```
    0x6e17, 0x7e36, 0x4e55, 0x5e74, 0x2e93, 0x3eb2, 0x0ed1, 0x1ef0
};

unsigned int crc_cal_by_byte(unsigned char* ptr, unsigned char len)
{
    unsigned short crc = 0xffff;

    while(len-- != 0)
    {
        unsigned int high = (unsigned int)(crc/256);
        crc <<= 8;
        crc ^= crc_ta_8[high^*ptr];
        ptr++;
    }

    return crc;
}
```

##

测试代码:

##

```
void main()
{
    unsigned char sample_data[] = {0x01, 0x01, 0x01, 0x06, 0xd9, 0xfc, 0x8c, 0x02, 0x01, 0x00,
0x01};//Result should be: 0x9b94
    unsigned char data1[] = {0x63};//Result should be: 0xbd35
    unsigned char data2[] = {0x8c};//Result should be: 0xb1f4
    unsigned char data3[] = {0x7d};//Result should be: 0x4eca
    unsigned char data4[] = {0xaa, 0xbb,0xcc};//Result should be: 0x6cf6
    unsigned char data5[] = {0x00,0x00,0xaa, 0xbb, 0xcc};//Result should be: 0xb166
    unsigned short r1 = 0, r2=0, r3=0, r4=0, r5=0, r_sample_data;

    //Implementation 1
    r1 = crc_cal_by_byte(data1, 1);
    r2 = crc_cal_by_byte(data2, 1);
    r3 = crc_cal_by_byte(data3, 1);
    r4 = crc_cal_by_byte(data4, 3);
    r5 = crc_cal_by_byte(data5, 5);
    r_sample_data = crc_cal_by_byte(sample_data, 11);
    printf("Implementation_1: r1= %x, r2=%x, r3=%x, r4=%x, r5=%x, r_sample_data=%x\n", r1, r2, r3,
r4, r5,r_sample_data);
    r1=r2=r3=r4=r5=0;
```

```
    //Implementation 2
    r1 = crc_cal_by_bit(data1, 1);
    r2 = crc_cal_by_bit(data2, 1);
    r3 = crc_cal_by_bit(data3, 1);
    r4 = crc_cal_by_bit(data4, 3);
    r5 = crc_cal_by_bit(data5, 5);
    r_sample_data = crc_cal_by_bit(sample_data, 11);
    printf("Implementation_2: r1= %x, r2=%x, r3=%x, r4=%x, r5=%x, r_sample_data=%x\n", r1, r2, r3,
r4, r5,r_sample_data);
    r1=r2=r3=r4=r5=0;

    //Implementation 3
    r1 = crc_cal_by_halfbyte(data1, 1);
    r2 = crc_cal_by_halfbyte(data2, 1);
    r3 = crc_cal_by_halfbyte(data3, 1);
    r4 = crc_cal_by_halfbyte(data4, 3);
    r5 = crc_cal_by_halfbyte(data5, 5);
    r_sample_data = crc_cal_by_halfbyte(sample_data, 11);
    printf("Implementation_3: r1= %x, r2=%x, r3=%x, r4=%x, r5=%x, r_sample_data=%x\n", r1, r2, r3,
r4, r5,r_sample_data);
    r1=r2=r3=r4=r5=0;
}
##############################################################
```

IMPORTANT NOTICE – PLEASE READ CAREFULLY

Hypersen Technologies Co., Ltd. reserve the right to make changes, corrections, enhancements, modifications, and improvements to Hypersen products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on Hypersen products before placing orders. Hypersen products are sold pursuant to Hypersen's terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of Hypersen products and Hypersen assumes no liability for application assistance or the design of Purchasers' products.

No license, express or implied, to any intellectual property right is granted by Hypersen herein.

Resale of Hypersen products with provisions different from the information set forth herein shall void any warranty granted by Hypersen for such product.

Hypersen and the Hypersen logo are trademarks of Hypersen. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.