

P 系列激光位移传感器

MODBUS 通信协议

Version 1.0.0.6

2025 年 03 月 03 日

变更记录

日期	版本	变更内容
2022 年 4 月 8 日	V1.0.0.0	发版
2022 年 8 月 22 日	V1.0.0.1	修改传感器系列名称
2022 年 9 月 19 日	V1.0.0.2	(1) 增加抽样数据上报功能； a. 参数设置类寄存器增加数据抽样比列、抽样方式设置寄存器； b. 数据读取类寄存器增加抽样数据：距离 1、距离 2、厚度等寄存器。 (2) 增加网络参数复位功能；
2023 年 2 月 23 日	V1.0.0.3	(1) 增加数据缓存功能； (2) 增加时域抽样数据复位功能； (对应固件版号：V1.1.16 及以上)
2023 年 8 月 23 日	V1.0.0.4	增加 MODBUS TCP 报文格式说明
2023 年 2 月 6 日	V1.0.0.5	(1) 增加位置 2、厚度置零寄存器； (2) 增加软重启功能；
2025 年 3 月 3 日	V1.0.0.6	(1) 参数设置类新增寄存器： a、位置 1（厚度）的修正斜率与偏置； b、modbus 从机地址与波特率。 (2) 数据读取类新增寄存器： a、激光温度； b、IP 地址； c、当前绑定主机的网络端口

目录

1. 引言	5
2. 通信规格	6
3. Modbus 通信规约	7
3.1. RTU 模式信息帧格式	7
3.2. TCP 模式信息帧格式	8
3.3. 传感器支持的功能码	10
3.3.1. 功能码 03 (0x03)	10
3.3.2. 功能码 04 (0x04)	11
3.3.3. 功能码 06 (0x06)	12
3.4. 错误响应帧格式	12
4. Modbus 寄存器分配	14
4.1. 参数设置类寄存器	14
4.1.1. 寄存器列表	14
4.1.2. 寄存器说明	16
4.2. 数据读取类寄存器	22
4.2.1. 寄存器列表	22
4.2.2. 寄存器说明	23
5. 应用示例	26
5.1. 读取参数示例（功能码 03）	26
5.1.1. ModbusRTU 格式数据解析	26
5.1.2. ModbusTCP 格式数据解析	27
5.2. 设置参数示例（功能码 06）	28
5.2.1. ModbusRTU 格式数据解析	28
5.2.2. ModbusTCP 格式数据解析	29
5.3. 读取数据示例（功能码 04）	30
5.3.1. ModbusRTU 格式数据解析	30
5.3.2. ModbusTCP 格式数据解析	31
5.4. 读取缓存数据示例（功能码 04）	33
6. 附录	35

附录 1 Modbus RTU CRC 算法代码	35
附录 2 Modbus 通信参数设置	37
附录 3 数据缓存功能说明	37

1. 引言

本文档介绍了如何使用 Modbus RTU 协议，实现对 TS-LTP-E 系列激光位移传感器的参数配置与数据读取功能。

使用前，注意事项：

- （1）使用 Modbus RTU 协议时，传感器只能作为从机使用。
- （2）使用前，请确保从机地址、串口波特率设置正确，从机地址取值范围：1 ~ 247。
- （3）从机地址、串口波特率可通过配套上位机软件进行设置，串口波特率设置最大不超过 115200 bits/s。（**默认设置：**从机地址为 **1**，波特率为 **115200** bits/s）
- （4）传感器从上电到完成配置，大约需要 5s。在配置完成后，才可通过 Modbus RTU 协议读写寄存器数据，否则读写操作会超时。

为方便操作，推荐使用配套的上位机软件完成参数设置。

Modbus 通信参数设置操作，见“附录 2 Modbus 通信参数设置”。

2. 通信规格

按照表 2-1 规格，进行 MODBUS 串行通信，波特率通过配套的上位机软件进行设置。

表 2-1 通信规格

通信规格	通信接口	RS-485，半双工 2 线制
	波特率	9600/19200/38400/57600/115200
	数据位	8 位
	奇偶校验位	无
	停止位	1 位
	通信协议	MODBUS RTU

按照表 2-2 规格，进行 MODBUS TCP 通信，IP 地址通过配套的上位机软件进行设置。

表 2-2 MODBUS TCP 通信规格

通信规格	通信接口	TCP/IP
	监听端口	502
	IP 地址	192.168.0.10（默认，可通过上位机软件更改）

3. Modbus 通信规约

3.1. RTU 模式信息帧格式

表 3-1 RTU 模式信息帧格式

开始	地址字段	功能代码	数据	错误校验 (CRC)	结束（起始）
3.5 个字符时间以上的间隔	1 个字节	1 个字节	0~252 个字节	2 字节	3.5 个字符时间以上的间隔

(1) 地址字段

地址字段为从设备地址，地址取值范围：0~247（十进制）。单个设备的实际地址范围是 1~247，地址 0 作为广播地址。主设备发送消息时，将从设备地址放到信息帧中，以便从设备识别此消息是否是发给本机的，如是本机地址，则从设备做出响应。从设备回复主设备时，在回应消息的地址域中提供本机地址，以便主设备识别是哪个从设备返回的数据。

(2) 功能码

功能码用于表示消息帧的功能，取值范围为 1~255（十进制）。从设备根据功能码，执行相应的功能，执行完成后在响应消息帧中返回同样的功能码。如果出现异常，返回的消息帧中将功能码最高位（MSB）设置为 1。

(3) 数据域

数据域存放功能码需要操作的具体数据。数据域以字节为单位，长度可变。

(4) 错误校验（CRC）

参与 CRC 计算的数据区域：包含一帧内从地址字段至数据区段内容，即计算 CRC 校验前的所有字节。

在 Modbus RTU 串行通信中，错误校验码是 16 位（2 个字节）的二进制值，采用 CRC-16 校验方法，多项式为 CRC-16/MODBUS: $x^{16} + x^{15} + x^2 + 1$ 。

错误校验码在数据帧中的传输顺序为：低字节在前，高字节在后。

CRC 值由发送设备计算，并添加到报文中。接收设备在报文接收过程中重新计算 CRC，并和接收的实际值进行比较,进行比较的值如果不同则为出错。

CRC 具体算法，见附录 1：Modbus RTU CRC 算法代码。

3.2. TCP 模式信息帧格式

Modbus TCP 数据帧与 Modbus RTU 协议数据帧的区别：

Modbus TCP 协议是在 RTU 协议前面添加 MBAP 报文头，由于 TCP 是基于可靠连接的服务，RTU 协议中的 CRC 校验码在此不需要，因此在 Modbus TCP 协议中没有 CRC 校验码；同时 RTU 报文中的“地址字段”被“单元标识符”替代，放在 Modbus TCP 应用协议报文头中。

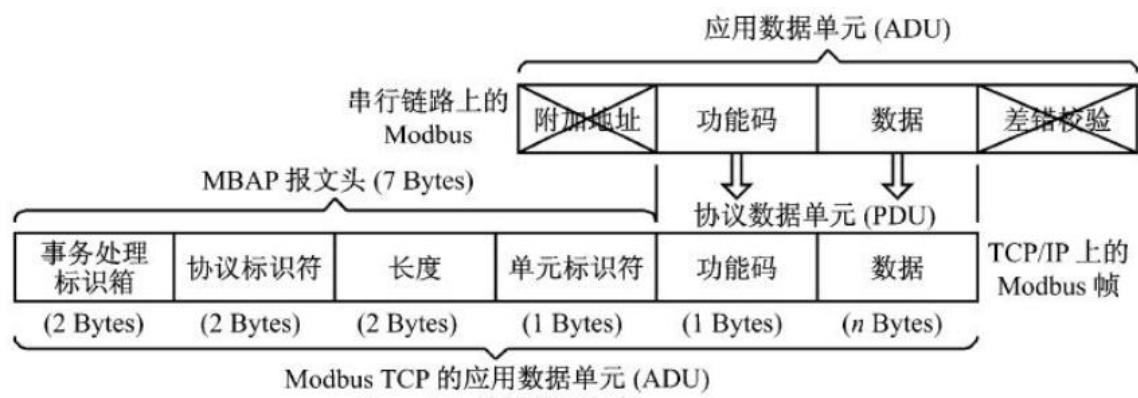


表 3-2 TCP 模式信息帧格式

事务处理标识	协议标识	长度	单元标识符	功能码	数据
2 个字节	2 个字节	2 个字节	1 个字节	1 个字节	0~253 个字节

Modbus TCP 协议数据帧由 MBAP (Modbus Application Protocol Head) 和 PDU (Protocol Data Unit) 两部分构成。

MBAP 报文头：事务处理标识 + 协议标识 + 长度 + 单元标识符；

PDU：功能码 + 数据；

表 3-3 MBAP 报文头说明

MBAP 报文头	事务处理标识	协议标识	长度	单元标识符
	2 个字节	2 个字节	2 个字节	1 个字节

(1) 事务处理标识

该标识可以理解为报文的序列号，一般每次通信之后就要加 1，用于区分不同的通信数据报文。

从站（服务器端）应答时从接收的请求中复制该值。

(2) 协议标识符

Modbus TCP 协议时，该值固定为 0x0000;

(3) 长度

用于表示后续字节数，即从单元标识符开始后面数据的长度（单元标识符+功能码+数据的总字节数），后续字节必须在 256 字节以内。

(4) 单元标识符

串口链路或其它总线上的远程终端标识，可以理解为设备地址。

从站（服务器端）应答时，从接收的请求中复制该值。

注：Modbus 设备可分为主站和从站。主站只有一个，从站有多个，主站向各从站发送请求帧，从站给予响应。在使用 TCP 通信时，主站为 client 端，主动建立连接；从站为 server 端，等待连接。

3.3. 传感器支持的功能码

本传感器支持的功能码，如表 3-4 所示：

表 3-4 支持的功能码列表

功能代码	功能名	操作
03H	读保持寄存器	读取 1 个或多个保持寄存器的内容
04H	读输入寄存器	读取 1 个或多个输入寄存器的内容
06H	写单个寄存器	向 1 个保持寄存器写入值

3.3.1. 功能码 03(0x03)

功能码 03 的请求帧数据格式（主机—>从机）如表 3-5 所示：

表 3-5 请求帧数据格式（功能码 03）

字节序号	0	1	2	3	4	5	6	7
数据内容	从机地址	功能码	起始地址 高字节	起始地址 低字节	寄存器数量 高字节	寄存器数量 低字节	CRC 校验低字节	CRC 校验高字节
取值范围	1~247	03	0 ~ 65535		1~125			

功能码 03 的响应帧数据格式（从机—>主机）如表 3-6 所示：

表 3-6 响应帧数据格式（功能码 03）

字节序号	0	1	2	3	4
数据内容	从机地址	功能码	字节数	寄存器值 1 高字节	寄存器值 1 低字节
取值范围	1~247	03	2*N (N 为读取的寄存器个数)			

字节序号	2N+1	2N+2	2N+3	2N+4
数据内容	寄存器值 N 高字节	寄存器值 N 低字节	CRC 校验 低字节	CRC 校验 高字节

取值范围				
------	--	--	--	--

3.3.2. 功能码 04(0x04)

功能码 04 的请求帧数据格式（主机—>从机）如表 3-7 所示：

表 3-7 请求帧数据格式（功能码 04）

字节序号	0	1	2	3	4	5	6	7
数据内容	从机地址	功能码	起始地址高字节	起始地址低字节	寄存器数量高字节	寄存器数量低字节	CRC 校验低字节	CRC 校验高字节
取值范围	1~247	04	0 ~ 65535		1~125			

功能码 04 的请求帧数据格式（从机—>主机）如表 3-8 所示：

表 3-8 响应帧数据格式（功能码 04）

字节序号	0	1	2	3	4
数据内容	从机地址	功能码	字节数	寄存器值 1 高字节	寄存器值 1 低字节
取值范围	1~247	04	2 * N (N 为读取的寄存器个数)			

字节序号	2N+1	2N+2	2N+3	2N+4
数据内容	寄存器值 N 高字节	寄存器值 N 低字节	CRC 校验高字节	CRC 校验低字节
取值范围				

3.3.3. 功能码 06(0x06)

功能码 06 的请求帧数据格式（主机—>从机）如表 3-9 所示：

表 3-9 请求帧数据格式（功能码 06）

序号	0	1	2	3	4	5	6	7
数据内容	从机地址	功能码	寄存器地址高字节	寄存器地址低字节	寄存器值高字节	寄存器值低字节	CRC 校验低字节	CRC 校验高字节
取值	1~247	06						

功能码 06 的响应帧数据格式（从机—>主机）如表 3-10 所示：

表 3-10 响应帧数据格式（功能码 06）

序号	0	1	2	3	4	5	6	7
数据内容	从机地址	功能码	寄存器地址高字节	寄存器地址低字节	寄存器值高字节	寄存器值低字节	CRC 校验低字节	CRC 校验高字节
取值	1~247	06						

3.4. 错误响应帧格式

当从机收到主机发送的信息帧时，先进行 CRC 校验，如 CRC 校验错误，则忽略该帧；

如从机检测到了 CRC 校验以外的错误，则向主机返回错误信息帧。错误信息帧中返回的功能码，是将主机发送的功能码最高位置 1 而来，相当于在主机发送的功能码基础上加 0x80。

举例：主机发送的功能码为 0x03，则对应的错误功能码为 0x83；

表 3-11 错误响应帧格式

字节序号	0	1	2	3	4
数据内容	地址码	功能码	错误码	CRC 校验低字节	CRC 校验高字节

表 3-12 错误码说明

错误码	错误码含义	详细说明
01	非法的功能码	接收到的功能码不支持

02	非法的数据地址	指定的地址超出从机的范围
03	非法的数据值	接收到主机发送的数据值超出相应地址的数据范围

4. Modbus 寄存器分配

4.1. 参数设置类寄存器

参数存放在保持寄存器（Holding Register）中。

参数设置与读取，通过功能码“03”、功能码“06”操作保持寄存器实现。

4.1.1. 寄存器列表

表 4-1 保持寄存器列表（对应功能码 03H、06H）

寄存器地址	功能名称	数据类型	寄存器数
0000	激光启停	2 字节 无符号	1
0001	光功率自动开关	2 字节 无符号	1
0002	光功率值设置	2 字节 无符号	1
0003	中值滤波参数设置	2 字节 无符号	1
0004	滑动平均滤波参数设置： 窗口尺寸	2 字节 无符号	1
0005	高/低通滤波器参数设置： 高/低通滤波器选择、滤波器频率	2 字节 无符号	1
0006	采样率设置	2 字节 无符号	1
0007	设置当前位置 1 为零点	2 字节 无符号	1
0008	无效数据保持点数设置	2 字节 无符号	1
0009	统计 1 启停	2 字节 无符号	1
0010	统计 1 参数： 数据源、统计类型	2 字节 无符号	1
0011	统计 1 复位	2 字节 无符号	1
0012	统计 2 启停	2 字节	1

		无符号	
0013	统计 2 参数： 数据源、统计类型	2 字节 无符号	1
0014	统计 2 复位	2 字节 无符号	1
0015	时间码复位	2 字节 无符号	1
0016	编号复位	2 字节 无符号	1
0017	时域数据抽样方式	2 字节 无符号	1
0018	时域数据抽样比例	2 字节 无符号	1
0019	网络参数复位或设备重启	2 字节 无符号	1
0020	存储参数	2 字节 无符号	1
0021	时域数据抽样复位	2 字节 无符号	1
0022	缓存数据上报个数	2 字节 无符号	1
0023	缓存区上锁标志	2 字节 无符号	1
0024	缓存数据类型	2 字节 无符号	1
0025	设置当前位置 2 为零点	2 字节 无符号	1
0026	设置当前厚度为零点	2 字节 无符号	1
0027	位置 1 修正斜率（高 16 位）	2 字节 无符号	1
0028	位置 1 修正斜率（低 16 位）	2 字节 无符号	1
0029	位置 1 偏置（高 16 位）	2 字节 无符号	1
0030	位置 1 偏置（低 16 位）	2 字节 无符号	1

0031	modbus 从机地址设置	2 字节 无符号	1
0032	modbus 波特率设置	2 字节 无符号	1
0033	厚度修正斜率（高 16 位）	2 字节 无符号	1
0034	厚度修正斜率（低 16 位）	2 字节 无符号	1
0035	厚度偏置（高 16 位）	2 字节 无符号	1
0036	厚度偏置（低 16 位）	2 字节 无符号	1
0037 ~ 0059	预留		23

注：

- （1）传感器定义的保持寄存器最大数量为 60 个；
- （2）寄存器写入无效值时，原寄存器的值不发生改变；

4.1.2. 寄存器说明

表 4-2 寄存器说明列表

寄存器地址	功能名称	取值范围	数据类型	读/写	寄存器数
0000	激光启停	0：停止 1：启动	2 字节 无符号	读/写	1
0001	光功率 自动开关	0：手动 1：自动	2 字节 无符号	读/写	1
0002	光功率设置	5 ~ 1000 对应光功率： 0.5%~100%	2 字节 无符号	读/写	1
0003	中值滤波器宽度	0x00：无滤波 0x01：3 0x02：5 0x03：9 0x04：15 0x05：31 0x06：63	2 字节 无符号	读/写	1
0004	滑动平均参数设置： 窗口尺寸	0x00：1 0x01：4	2 字节 无符号	读/写	1

		0x02: 16 0x03: 64 0x04: 256 0x05: 1024			
0005	高低通滤波器参数设置: 高 8 位 表示滤波器选择; 低 8 位 表示高/低通滤波器频率	高 8 位参数: 0x00: 无滤波器 0x01: 低通 0x02: 高通 低 8 位参数: 0x00: 0.01fs 0x01: 0.02fs 0x02: 0.04fs 0x03: 0.10fs 0x04: 0.20fs 0x05: 0.40fs 例: 0x0103 表示使用低通滤波器, 滤波器频率 0.10fs。	2 字节 无符号	读/写	1
0006	采样率设置	0x08: 6.25us 0x09: 8us 0x0a: 10us 0x0b: 12.5us 0x0c: 16us 0x0d: 20us 0x0e: 25us 0x0f: 32us 0x10: 40us 0x11: 50us 0x12: 62.5us 0x13: 80us 0x14: 100us 0x15: 125us 0x16: 160us 0x17: 200us 0x18: 250us 0x19: 320us	2 字节 无符号	读/写	1

		0x1a: 400us 0x1b: 500us 0x1c: 625us 0x1d: 800us 0x1e: 1ms			
0007	设置当前位置 1 为零点	0: 取消归零,距离的映射偏置会置为 0; 1: 位置归零,将当前位置 1 设置为零点,可重复归零。	2 字节 无符号	读/写	1
0008	无效数据保持点数设置	0 ~ 65535 注: 值为 65535 时,表示一直保持最后 1 次有效值;	2 字节 无符号	读/写	1
0009	统计 1 启停	0: 统计停止 1: 统计使能	2 字节 无符号	读/写	1
0010	统计 1 参数: 高 8 位: 数据源; 低 8 位: 统计类型;	高 8 位: 0: 距离 1 1: 距离 2 2: 厚度 低 8 位: 0: 最大值 1: 最小值 2: 峰峰值 例: 0x0102 表示距离 2 的峰峰值; 0x0000 表示距离 1 的最大值。	2 字节 无符号	读/写	1
0011	统计 1 复位	1: 统计复位; 其它值无效	2 字节 无符号	读/写	1
0012	统计 2 启停	0: 统计停止 1: 统计使能	2 字节 无符号	读/写	1
0013	统计 2 参数:	高 8 位:	2 字节	读/写	1

	高 8 位：数据源； 低 8 位：统计类型；	0：距离 1 1：距离 2 2：厚度 低 8 位： 0：最大值 1：最小值 2：峰峰值 例： 0x0102 表示距离 2 的峰峰值； 0x0000 表示距离 1 的最大值。	无符号		
0014	统计 2 复位	1：统计复位； 其它值无效	2 字节 无符号	读/写	1
0015	时间码复位	1：时间码复位； 其它值无效	2 字节 无符号	读/写	1
0016	编号复位	1：编号复位； 其它值无效	2 字节 无符号	读/写	1
0017	时域数据抽样方式	0：不抽样； 1：最近相邻点抽样； 2：平均值抽样；	2 字节 无符号	读/写	1
0018	时域数据抽样比例	取 值 范 围： 1~65535	2 字节 无符号	读/写	1
0019	网络参数复位或设备重启	1：网络参数复位； 170：设备软件重启； 其它值无效	2 字节 无符号	读/写	1
0020	存储参数	1：将当前参数写入到 flash，存储结束后，该值自动置为 0； 其它设置值无效。	2 字节 无符号	读/写	1
0021	时域数据抽样复位	1：抽样数据复位，抽样重新统	2 字节 无符号	读/写	1

		计; 其它设置值无效。			
0022	缓存数据上报个数	缓存数据需上报的总个数, 取值范围: 0 ~ 50000, 取值为 0 时, 不进行数据缓存。	2 字节 无符号	读/写	1
0023	缓存区上锁标志	0: 缓存区解锁, 新数据写入缓存区, 旧缓存数据清 0; 1: 缓存区上锁, 新数据不写入缓存区; 其它值无效。	2 字节 无符号	读/写	1
0024	缓存数据类型	0: 距离 1; 1: 距离 2; 2: 厚度;	2 字节 无符号	读/写	1
0025	设置当前位置 2 为零点	0: 取消归零,; 1: 归零, 将当前位置 2 数据设置为零点, 可重复归零	2 字节 无符号	读/写	1
0026	设置当前厚度为零点	0: 取消归零 1: 归零, 将当前厚度值设置为零点, 可重复归零。	2 字节 无符号	读/写	1
0027	位置 1 修正斜率 (高 16 位)	见后续注释	2 字节 无符号	读/写	1
0028	位置 1 修正斜率 (低 16 位)	见后续注释	2 字节 无符号	读/写	1
0029	位置 1 偏置 (高 16 位)	见后续注释	2 字节 无符号	读/写	1
0030	位置 1 偏置 (低 16 位)	见后续注释	2 字节 无符号	读/写	1
0031	modbus 从机地址设置	1~247	2 字节	读/写	1

			无符号		
0032	modbus 波特率设置	0: 9600 1: 19200 2: 38400 3: 57600 4: 115200	2 字节 无符号	读/写	1
0033	厚度修正斜率（高 16 位）	见后续注释	2 字节 无符号	读/写	1
0034	厚度修正斜率（低 16 位）	见后续注释	2 字节 无符号	读/写	1
0035	厚度偏置（高 16 位）	见后续注释	2 字节 无符号	读/写	1
0036	厚度偏置（低 16 位）	见后续注释	2 字节 无符号	读/写	1

注：

- (1) 时域数据抽样方式或抽样比例变化时，Modbus 上报的抽样数据值会复位为 0。
- (2) 网络参数复位后的参数，IP 地址：192. 168. 0. 10，子网掩码：255. 255. 255. 0，网关：192. 168. 0. 1，复位后的参数未存入 flash。
- (3) 位置 1、位置 2、厚度三种数据，可分别进行置零，当位置 1、位置 2 零点关联开关打开时，位置 1、位置 2 使用同一个零点基准，关闭时，位置 1、位置 2 可分别进行置零。厚度置零，通常用于测量透明物体或对射测厚需要选择一个基准值时。
- (4) 位置 1（厚度）的修正斜率和位置 1（厚度）的偏置，设置时均需进行系数转换。修正斜率的转换系数为 1048576，即 $(1 \ll 20)$ ；偏置的转换系数为 1000000。

位置 1（厚度）的修正斜率与偏置，正确的设置顺序为：先设置低 16 位，再设置高 16 位。

例：设置位置 1 修正斜率为 2，位置 1 斜率偏置为 0.5。

1) 设置位置 1 修正斜率为 2：

- ① 参数转换： $2 * 1048576 = 2097152$ 转成 0x0020 0000（十六进制）；
- ② 设置低位值：即寄存器 28 设置为 0x0000；
- ③ 设置高位值：即寄存器 27 设置为 0x0020。

2) 设置位置 1 斜率偏置为 0.5：

- ① 参数转换： $0.5 * 1000000 = 500000$ 转成 0x0007 A120（十六进制）；
- ② 设置低位值：即寄存器 30 设置为 0xA120；
- ③ 设置高位值：即寄存器 29 设置为 0x0007。

4.2. 数据读取类寄存器

测量数据存放在输入寄存器中（Input Register）。

通过功能码“04”操作输入寄存器，实现测量数据的读取操作。

4.2.1. 寄存器列表

表 4-3 输入寄存器列表（对应功能码 04H）

寄存器地址	功能名称	单位	数据类型	寄存器数	备注
0000 ~ 0001	距离 1	mm	4 字节 有符号	2	数值扩大了 1000000 倍
0002 ~ 0003	距离 2	mm	4 字节 有符号	2	数值扩大了 1000000 倍
0004 ~ 0005	厚度	mm	4 字节 有符号	2	数值扩大了 1000000 倍
0006 ~ 0007	统计值 1	mm	4 字节 有符号	2	数值扩大了 1000000 倍
0008 ~ 0009	统计值 2	mm	4 字节 有符号	2	数值扩大了 1000000 倍
0010 ~ 0011	抽样数据： 距离 1	mm	4 字节 有符号	2	数值扩大了 1000000 倍
0012 ~ 0013	抽样数据： 距离 2	mm	4 字节 有符号	2	数值扩大了 1000000 倍
0014 ~ 0015	抽样数据： 厚度	mm	4 字节 有符号	2	数值扩大了 1000000 倍
0016	缓存数据已读 个数	个	2 字节 无符号	2	
0017	本次缓存数据 上报的有效个 数	个	2 字节 无符号	2	
0018 ~ 0019	缓存数据 1	mm	4 字节 有符号	2	数值扩大了 1000000 倍
0020 ~ 0021	缓存数据 2	mm	4 字节 有符号	2	数值扩大了 1000000 倍
.....					
0136 ~ 0137	缓存数据 60	mm	4 字节	2	数值扩大了

			有符号		1000000 倍
0138 ~ 0140	预留			3	
0140	激光温度	℃	2 字节 无符号	1	
0141 ~ 0149	预留			9	
0150	IP 地址 (第 1 位)		2 字节 无符号	1	
0151	IP 地址 (第 2 位)		2 字节 无符号	1	
0152	IP 地址 (第 3 位)		2 字节 无符号	1	
0153	IP 地址 (第 4 位)		2 字节 无符号	1	
0154	当前绑定主机的 网络端口		2 字节 无符号	1	
0155 ~ 0159	预留			5	

4.2.2. 寄存器说明

表 4-4 输入寄存器说明

寄存器地址	功能名称	数据类型	读/写	寄存器个数
0000	距离 1 高 16 位	2 字节 有符号	只读	1
0001	距离 1 低 16 位	2 字节 有符号	只读	1
0002	距离 2 高 16 位	2 字节 有符号	只读	1
0003	距离 2 低 16 位	2 字节 有符号	只读	1
0004	厚度高 16 位	2 字节 有符号	只读	1
0005	厚度低 16 位	2 字节 有符号	只读	1

0006	统计值 1 高 16 位	2 字节 无符号	只读	1
0007	统计值 1 低 16 位	2 字节 无符号	只读	1
0008	统计值 2 高 16 位	2 字节 无符号	只读	1
0009	统计值 2 低 16 位	2 字节 无符号	只读	1
0010	抽样数据： 距离 1 高 16 位	2 字节 无符号	只读	1
0011	抽样数据： 距离 1 低 16 位	2 字节 无符号	只读	1
0012	抽样数据： 距离 2 高 16 位	2 字节 无符号	只读	1
0013	抽样数据： 距离 2 低 16 位	2 字节 无符号	只读	1
0014	抽样数据： 厚度高 16 位	2 字节 无符号	只读	1
0015	抽样数据： 厚度低 16 位	2 字节 无符号	只读	1
0016	缓存数据 已读个数	2 字节 无符号	只读	1
0017	本次缓存数据上 报的有效个数	2 字节 无符号	只读	1
0018	缓存数据 1 高 16 位	2 字节 无符号	只读	1
0019	缓存数据 1 低 16 位	2 字节 无符号	只读	1
0020	缓存数据 2 高 16 位	2 字节 无符号	只读	1
0021	缓存数据 2 低 16 位	2 字节 无符号	只读	1

.....
0136	缓存数据 60 高 16 位	2 字节 无符号	只读	1
0137	缓存数据 60 低 16 位	2 字节 无符号	只读	1
.....
0140	激光温度	2 字节 无符号	只读	1
.....
0150	IP 地址 (第 1 位)	2 字节 无符号	只读	1
0151	IP 地址 (第 2 位)	2 字节 无符号	只读	1
0152	IP 地址 (第 3 位)	2 字节 无符号	只读	1
0153	IP 地址 (第 4 位)	2 字节 无符号	只读	1
0154	当前绑定主机的 网络端口	2 字节 无符号	只读	1
.....

注：

- (1) 传感器定义的输入寄存器数量为 160 个，超出范围将会报错；
- (2) 时域数据抽样设置为无抽样或抽样比例为 1 时，上报的抽样数据是实时值；
- (3) 缓存数据类型由参数设置类寄存器“缓存数据类型”设置值决定；
- (4) 缓存数据上报时，每次上报的缓存数据个数为 60 个。
- (5) “采样间隔”、“抽样系数”、“缓存数据上报个数”、“缓存数据类型”等参数变更时，缓存数据的相关变量（缓存区上锁标志、缓存区数据）均会进行复位。

5. 应用示例

5.1. 读取参数示例（功能码 03）

5.1.1. ModbusRTU 格式数据解析

举例：传感器的从机地址为“1”,读取传感器通道 1 的光源开关、曝光模式、曝光时间等参数。

分析：读取传感器参数，需使用“功能码 03”，读取保持寄存器的值。通道 1 光源开关、通道 1 曝光模式、通道 1 曝光时间对应的寄存器地址分别为 0000、0001、0002，因此可通过一次性读取 3 个寄存器的方式读取参数值。即用功能码 03，读保持寄存器起始地址为 0000，读取寄存器个数为 3 个。

主机发送请求报文：01 03 00 00 00 03 05 CB

表 5-1 读取参数请求报文解析

主机发送报文	占字节数	数据内容（16 进制）
从机地址	1	01
功能码	1	03
寄存器起始地址	2	00 00
读取寄存器个数	2	00 03
CRC 校验	2	05 CB

从机发送响应报文：01 03 06 00 01 00 01 00 05 DC B6

表 5-2 读取参数响应报文解析

从机响应报文	占字节数	数据内容（16 进制）
从机地址	1	01
功能码	1	03
字节数	1	06
寄存器 0000 的数据	2	00 01
寄存器 0001 的数据	2	00 00

寄存器 0002 的数据	2	00 05
CRC 校验	2	DC B6

数值分析：光源开关打开、曝光模式为手动、光功率设手动定值为 0.5。

5.1.2. ModbusTCP 格式数据解析

主机发送请求报文：00 00 00 00 00 06 01 03 00 00 00 03

表 5-3 读取参数请求报文解析

主机发送报文	占字节数	数据内容（16 进制）
事务处理标识	2	00 00
协议标识	2	00 00
后续字节长度	2	00 06
单元标识符	1	01
功能码	1	03
寄存器起始地址	2	00 00
读取寄存器个数	2	00 03

传感器发送的响应报文：00 00 00 00 00 09 01 03 06 00 01 00 00 00 05

表 5-4 读取参数响应报文解析(TCP)

从机响应报文	占字节数	数据内容（16 进制）
事务处理标识	2	00 00
协议标识	2	00 00
后续字节长度	2	00 09
单元标识符	1	01
功能码	1	03

数据长度	1	06
寄存器 0000 的数据	2	00 01
寄存器 0001 的数据	2	00 00
寄存器 0002 的数据	2	00 05

数值分析：光源开关打开、曝光模式为手动、光功率设手动定值为 0.5。

5.2. 设置参数示例（功能码 06）

5.2.1. ModbusRTU 格式数据解析

举例：传感器的从机地址为“1”，设置传感器的采样间隔为 100us。

分析：设置采样间隔，需使用“功能码 06”，往地址为 0006 的保持寄存器写入值，采样间隔对应的寄存器值为 20（0x14），即向地址为“0006”的保持寄存器，写入 0x14 数值。

主机发送请求报文：01 06 00 06 00 14 69 C4

表 5-5 设置参数请求报文解析

主机发送报文	占字节数	数据内容（16 进制）
从机地址	1	01
功能码	1	06
寄存器地址	2	00 06
寄存器值	2	00 14
CRC 校验	2	69 C4

从机发送响应报文：01 06 00 06 00 14 69 C4

表 5-6 设置参数响应报文解析

从机响应报文	占字节数	数据内容（16 进制）
从机地址	1	01
功能码	1	06
寄存器地址	2	00

		06
寄存器值	2	00 14
CRC 校验	2	69 C4

5.2.2. ModbusTCP 格式数据解析

主机发送请求报文：00 00 00 00 00 06 01 06 00 06 00 14

表 5-7 读取参数请求报文解析（TCP）

主机发送报文	占字节数	数据内容（16 进制）
事务处理标识	2	00 00
协议标识	2	00 00
后续字节长度	2	00 06
单元标识符	1	01
功能码	1	06
寄存器起始地址	2	00 06
寄存器值	2	00 10

传感器发送的响应报文：00 00 00 00 00 06 01 06 00 06 00 14

表 5-8 读取参数响应报文解析（TCP）

从机响应报文	占字节数	数据内容（16 进制）
事务处理标识	2	00 00
协议标识	2	00 00
后续字节长度	2	00 06
单元标识符	1	01
功能码	1	06
寄存器起始地址	2	00 06

寄存器值	2	00 14
------	---	----------

5.3. 读取数据示例（功能码 04）

5.3.1. ModbusRTU 格式数据解析

举例：传感器的从机地址为“1”，读取该传感器的测量数据：距离 1、距离 2、厚度。

分析：读取距离 1，需使用“功能码 04”，读取输入寄存器地址 0x0000-0x0005 的数值，读取的寄存器个数为 6 个。

主机发送的请求报文：01 04 00 00 00 06 70 08

表 5-9 读取数据请求报文解析

主机发送报文	占字节数	数据内容（16 进制）
从机地址	1	01
功能码	1	04
寄存器起始地址	2	00 00
读取寄存器个数	2	00 06
CRC 校验	2	70 08

传感器发送的响应报文：01 04 0C 00 36 B0 BE 00 20 24 5C 00 56 D3 04 41 47

表 5-10 读取数据响应报文解析

从机响应报文	占字节数	数据内容（16 进制）
从机地址	1	01
功能码	1	04
字节数	1	0C
寄存器 0000 的数据	2	00 36
寄存器 0001 的数据	2	B0 BE
寄存器 0002 的数据	2	00 20
寄存器 0003 的数据	2	24

		5C
寄存器 0004 的数据	2	00 56
寄存器 0005 的数据	2	D3 04
CRC 校验	2	41 47

数值分析：

距离 1 的数值为 0x 0036B0BE，转换为有符号整数为 3584190，值放大了 1000000 倍，距离 1 的实际测量值为 3,584190mm；

距离 2 的数值为 0x 0020245C，转换为有符号整数为 2106460，值放大了 1000000 倍，距离 2 的实际测量值为 2.106460mm；

厚度的数值为 0x0056D304，转换为有符号整数为 5690116，值放大了 1000000 倍，厚度的实际测量值为 5.690116mm。

5.3.2. ModbusTCP 格式数据解析

主机发送请求报文：00 00 00 00 00 06 01 04 00 00 00 06

表 5-11 读取参数请求报文解析（TCP）

主机发送报文	占字节数	数据内容（16 进制）
事务处理标识	2	00 00
协议标识	2	00 00
后续字节长度	2	00 06
单元标识符	1	01
功能码	1	04
寄存器起始地址	2	00 00
读取寄存器个数	2	00 06

传感器发送的响应报文：00 00 00 00 00 0F 01 04 0C 00 35 CB 69 00 1F 3A 6C 00 2A 05 D5

表 5-12 读取参数响应报文解析(TCP)

从机响应报文	占字节数	数据内容（16 进制）
事务处理标识	2	00 00
协议标识	2	00 00
后续字节长度	2	00 0F
单元标识符	1	01
功能码	1	04
数据长度	1	0C
寄存器 0000 的数据	2	00 35
寄存器 0001 的数据	2	CB 69
寄存器 0002 的数据	2	00 1F
寄存器 0003 的数据	2	3A 6C
寄存器 0004 的数据	2	00 2A
寄存器 0005 的数据	2	05 D5

注：当读取的数据值为 0x7FFFFFFF（16 进制）或 2147483647（有符号整数）时，则表示当前的测量值无效。

5.4. 读取缓存数据示例（功能码 04）

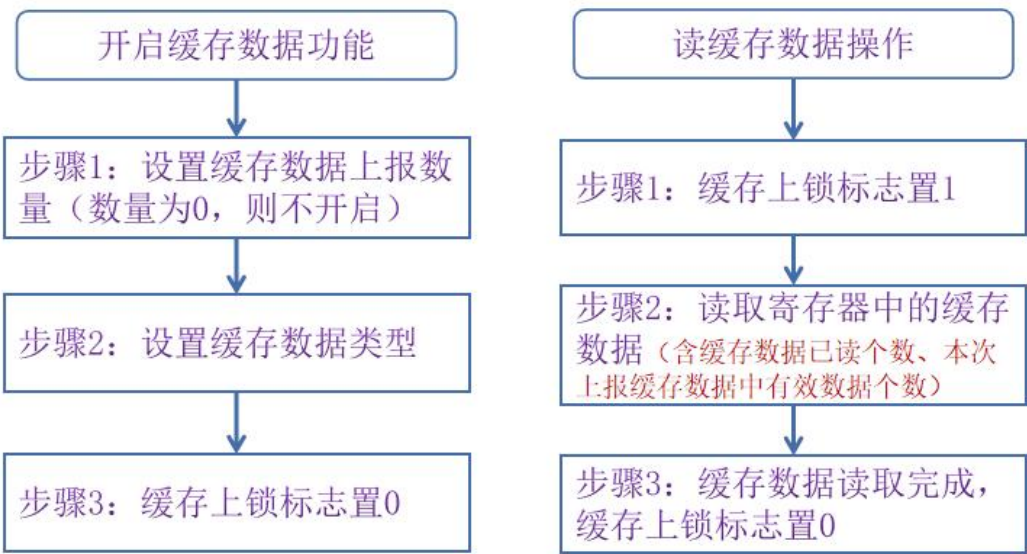


图 5-1 开启缓存数据功能与读缓存数据操作

举例：

使用外部触发功能，单次触发采集 1000 个距离 1 数据，数据采集结束后，通过 Modbus RTU 协议将 1000 个数据读出。即：读取距离 1 的缓存数据，缓存数据上报个数为 1000 个。

分析：

- (1) 读取距离 1 的缓存数据，需将“缓存数据类型”寄存器（24）值设置为 0，即距离 1；
- (2) 缓存数据上报个数为 1000 个，需将“缓存数据上报个数”寄存器（22）值设置为 1000；
- (3) 读缓存数据前，需将“缓存区上锁标志”寄存器（23）值设置为 1；
- (4) 由于 modbus 一帧最大只能读取 60 个缓存数据，即 120 个寄存器。为防止漏读数据，直接从“缓存数据已读个数”寄存器开始读取，连续读取 122 个寄存器：已读个数（1 个）+有效个数（1 个）+缓存数据（60*2 个）。
- (5) 1000 个缓存数据需要读取 17 次才能读完，每次固定读取 60 个缓存数据。前 16 次上报的缓存数据中有效个数为 60，第 17 次上报的缓存数据中有效个数为 40 个。
- (6) 读完 1000 个缓存数据后，将“缓存区上锁标志”寄存器（23）置 0。

操作步骤：

- 开启缓存数据功能操作：
 - ① “缓存数据类型” 设置为距离 1；
 - ② “缓存数据上报个数” 设置为 1000；
 - ③ “缓存区上锁标志” 设置为 0。

- 缓存数据读取操作：
- ① “缓存区上锁标志” 设置为 1；
 - ② 从“缓存数据已读个数” 寄存器开始读取 122 个寄存器，连续读 17 次。

➤ 缓存数据读完后操作：

“缓存区上锁标志” 设置为 0。

读取缓存数据，使用“功能码 04”，读取输入寄存器起始地址“16”（10 进制），读取的寄存器个数为 122 个。

主机发送的请求报文：01 04 00 10 00 7A 70 2C

表 5-13 读取缓存数据请求报文解析

主机发送报文	占字节数	数据内容（16 进制）
从机地址	1	01
功能码	1	04
寄存器起始地址	2	00 10
读取寄存器个数	2	00 7A
CRC 校验	2	70 2C

注：数据缓存功能的详细说明，见“附录 3 数据缓存功能说明”。

6. 附录

附录 1 Modbus RTU CRC 算法代码

```
static const unsigned char aucCRCHi[] = {  
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,  
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,  
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,  
    0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,  
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,  
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,  
    0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,  
    0x00, 0xC1, 0x81, 0x40  
};  
  
static const unsigned char aucCRCLo[] = {  
    0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06, 0x07, 0xC7,  
    0x05, 0xC5, 0xC4, 0x04, 0xCC, 0x0C, 0x0D, 0xCD, 0x0F, 0xCF, 0xCE, 0x0E,  
    0x0A, 0xCA, 0xCB, 0x0B, 0xC9, 0x09, 0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9,  
    0x1B, 0xDB, 0xDA, 0x1A, 0x1E, 0xDE, 0xDF, 0x1F, 0xDD, 0x1D, 0x1C, 0xDC,  
    0x14, 0xD4, 0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2, 0x12, 0x13, 0xD3,  
    0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3, 0xF2, 0x32,
```

```

0x36, 0xF6, 0xF7, 0x37, 0xF5, 0x35, 0x34, 0xF4, 0x3C, 0xFC, 0xFD, 0x3D,
0xFF, 0x3F, 0x3E, 0xFE, 0xFA, 0x3A, 0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38,
0x28, 0xE8, 0xE9, 0x29, 0xEB, 0x2B, 0x2A, 0xEA, 0xEE, 0x2E, 0x2F, 0xEF,
0x2D, 0xED, 0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6, 0x26,
0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60, 0x61, 0xA1,
0x63, 0xA3, 0xA2, 0x62, 0x66, 0xA6, 0xA7, 0x67, 0xA5, 0x65, 0x64, 0xA4,
0x6C, 0xAC, 0xAD, 0x6D, 0xAF, 0x6F, 0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB,
0x69, 0xA9, 0xA8, 0x68, 0x78, 0xB8, 0xB9, 0x79, 0xBB, 0x7B, 0x7A, 0xBA,
0xBE, 0x7E, 0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4, 0x74, 0x75, 0xB5,
0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71, 0x70, 0xB0,
0x50, 0x90, 0x91, 0x51, 0x93, 0x53, 0x52, 0x92, 0x96, 0x56, 0x57, 0x97,
0x55, 0x95, 0x94, 0x54, 0x9C, 0x5C, 0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E,
0x5A, 0x9A, 0x9B, 0x5B, 0x99, 0x59, 0x58, 0x98, 0x88, 0x48, 0x49, 0x89,
0x4B, 0x8B, 0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D, 0x4D, 0x4C, 0x8C,
0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42, 0x43, 0x83,
0x41, 0x81, 0x80, 0x40

```

```
};
```

```
unsigned short MB_CRC16(unsigned char * pucFrame, unsigned short usLen )
```

```
{
```

```
    unsigned char    crc_h = 0xFF;
```

```
    unsigned char    crc_l = 0xFF;
```

```
    unsigned short   index,   crc_value;
```

```
    while( usLen-- )
```

```
    {
```

```
        index = crc_l ^ *( pucFrame++ );
```

```
        crc_l = (unsigned char)(crc_h ^ aucCRCHi[index]);
```

```
        crc_h = aucCRCLo[index];
```

```
    }
```

```
    crc_value = crc_h;
```

```
    crc_value <<= 8;
```

```
    crc_value |= crc_l;
```

```
    return crc_value;
```

}

附录 2 Modbus 通信参数设置

Modbus 通信参数设置包括从机地址、波特率设置。参数设置步骤如下所示：

步骤 1： 打开上位机软件，与传感器成功连接；

步骤 2： 进入 Modbus 配置界面

在上位机软件主窗口，点击菜单栏“高级配置”——>通信配置——>Modbus 配置，如图 6-1 所示。

步骤 3： 设置从机地址

在从机地址设置框里选择输入设置的地址，取值范围 1-247；

步骤 4： 设置 Modbus 通信波特率

在波特率配置设置下拉框里选择要设置的波特率。



图 6-1 Modbus 通信参数配置

附录 3 数据缓存功能说明

数据缓存功能用于读取已设置缓存数据类型的多次测量结果，最大允许读取 2500 个缓存数据。modbus 单次读取的最大数据量为 60 个，当需上报的缓存数据个数超过 60 个后，则需多次读取。

缓存数据读取的时间顺序：先入先出。

注意： 读取缓存数据时，每次读取缓存数据的个数不能小于本次缓存数据上报的有效个数，否则会丢失数据。

例： 假设“数据缓存上报个数”设置值为 610，实际缓存个数大于 610 个。

则 modbus 读取缓存数据时需读取 11 次，前 10 次每次读取缓存数据个数应为 60 个，否则会漏

读数据。

数据缓存参数说明：

(1) 缓存数据上报个数

该参数用于设置可读取的最大缓存数据个数。该参数值设置为非 0 值时，传感器会启动数据缓存功能。

例：“数据缓存上报个数”设置值为 2000，则在实际数据缓存已满 2000 个时，此时去读缓存数据，则需连续读 34 次，才能将缓存数据读完。前 33 次，每次必须读取 60 个缓存数据，第 34 次读取 $(2000-60\times 33)=20$ 个缓存数据。

(2) 缓存区上锁标志

缓存区上锁标志用于对数据缓存区进行上锁操作。

当要读取缓存区数据时，必须先将“缓存区上锁标志”置 1，相当于对缓存区上锁，此时新数据不写入缓存区；在缓存数据读取完成后，必须将“缓存区上锁标志”置 0，相当于对缓存区解锁，此时新数据会写入缓存区。

(3) 缓存数据已读个数

该值用于指示数据缓存区锁定（缓存区上锁标志“置 1”）后，从缓存区已读取的有效数据总个数。

例：“数据缓存上报个数”设置值为 2000，在缓存数据读取时刻，缓存区当前只缓存了 1000 个，则在缓存区缓存数据全部读取完成后，“缓存数据已读个数”值为 1000。

(4) 本次缓存数据上报的有效个数

该值用于指示本次 modbus 60 个缓存数据寄存器中有效的数据个数。

例 1：“数据缓存上报个数”设置值为 60，而当前实际只缓存了 20 个，则“本次缓存数据上报的有效个数”值为 20。

例 2：“数据缓存上报个数”设置值为 2000 且实际数据缓存已满 2000 个，则需连续读 34 次数据，才能将缓存数据读完。

每次读取的缓存数据个数如下：第 1 到第 33 次，每次必须读取 60 个缓存数据，第 34 次读取 $(2000 - 60\times 33) = 20$ 个缓存数据。也就是“本次缓存数据上报的有效个数”寄存器前 33 次对应的值为 60，第 34 次对应的值为 20。

(5) 数据缓存时刻

指采集数据写入缓存区的时刻，由“抽样比例”决定。

例：当“抽样比例”设置为 1 时，则对每次采集的数据进行缓存；当“抽样比例”设置为

10 时，则每采集到 10 个数据，就将第 10 个数据写入缓存。

注：

- ① “采样间隔”与“抽样比例”参数修改时，缓存数据会进行复位；
- ② “抽样比例”参数，可通过对应的上位机软件进行设置。

