

P□601	RS-485通讯轴地址		速度	位置	扭矩
	设定范围	设定单位	出厂设定	电源重起	
	1 ~ 127	——	1 (A轴) 2 (b轴)	需要	
P□602	RS-485通讯超时		速度	位置	扭矩
	设定范围	设定单位	出厂设定	电源重起	
	0 ~ 1000	100ms	0	不需要	
<div><div></div><div><ul style="list-style-type: none">• P□602设置为零时，关闭通讯超时检测；• P□602设置为大于零时，表示必须在设定的时间内通讯，否则将出现通讯错误。举例，P□602设置成50时，表示必须每5秒与伺服驱动器通讯1次。</div></div>					

6.3 MODBUS 通讯协议

使用 RS-485 通讯时，每一台伺服驱动器必须预先设置参数 P□600 ~ P□601。通讯方式使用 MODBUS 协议，其中可使用下列两种模式：

ASCII 模式

RTU 模式。

以下说明 MODBUS 通讯。

■ 编码意义

ASCII 模式：

每个 8-bit 数据由两个 ASCII 字符组成。例如：一个 1-byte 数据 64_H（十六进制表示法）。以 ASCII 码“64”表示，包含了‘6’的 ASCII 码（36_H）和‘4’的 ASCII 码（34_H）。数字 0 至 9、字母 A 至 F 的 ASCII 码，如下表：

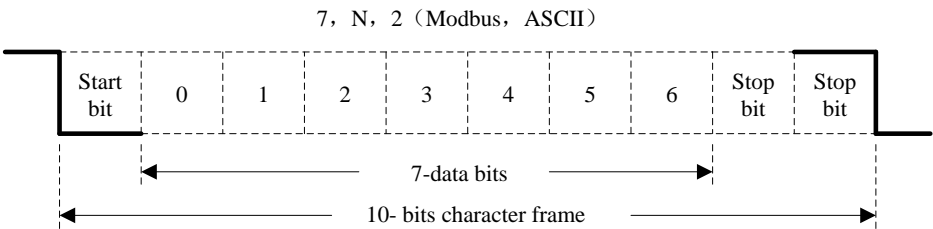
字符符号	‘0’	‘1’	‘2’	‘3’	‘4’	‘5’	‘6’	‘7’
对应 ASCII 码	30 _H	31 _H	32 _H	33 _H	34 _H	35 _H	36 _H	37 _H
字符符号	‘8’	‘9’	‘A’	‘B’	‘C’	‘D’	‘E’	‘F’
对应 ASCII 码	38 _H	39 _H	41 _H	42 _H	43 _H	44 _H	45 _H	46 _H

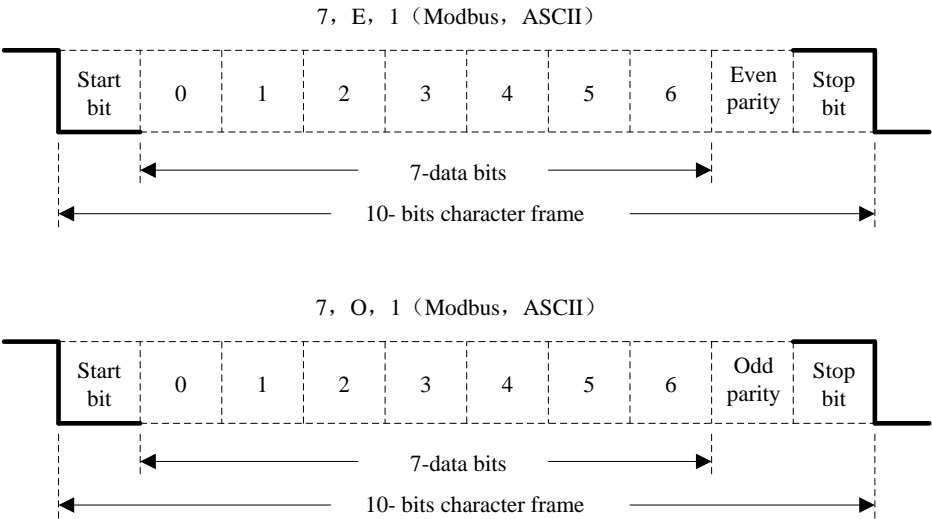
RTU 模式：

每个 8-bit 数据由两个 4-bit 的十六进制数据组成。例如：十进制 100 用 1-byte 的 RTU 数据表示为 64_H。

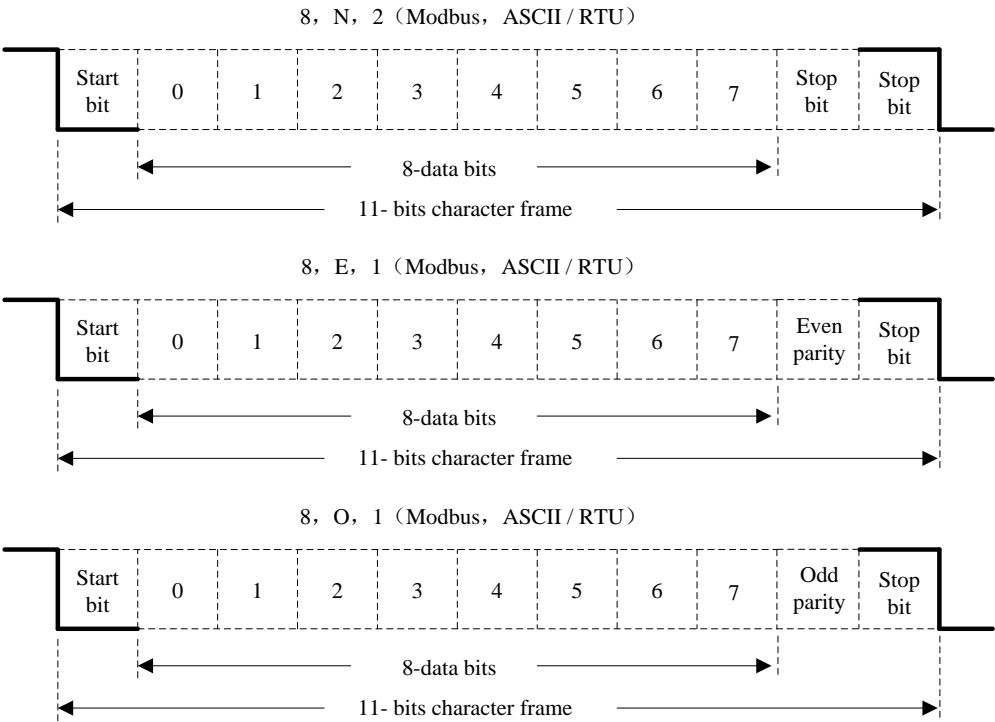
■ 字符结构

10bit 字符格式（用于 7-bit 数据）





11bit 字符格式（用于 8-bit 数据）



■ 通讯数据结构

通讯数据结构：

ASCII 模式：

STX	起始字符 ‘:’ => (3A _H)
ADR	通讯地址=>1-byte 包含了 2 个 ASCII 码
CMD	指令码=>1-byte 包含了 2 个 ASCII 码
DATA(n-1)	数据内容=>n-word=2n-byte 包含了 4n 个 ASCII 码, n 不大于 12
DATA(0)	
LRC	校验码=>1-byte 包含了 2 个 ASCII 码

End 1	结束码 1=> (0D _H) (CR)
End 0	结束码 0=> (0A _H) (LF)

RTU 模式:

STX	至少 4 个字节传输时间的静止时段
ADR	通讯地址=>1-byte
CMD	指令码=>1-byte
DATA(n-1)	数据内容=>n-word=2n-byte, n 不大于 12
.....	
DATA(0)	
CRC	CRC 校验码=>1-byte
End 1	至少 4 个字节传输时间的静止时段

通讯协议的数据格式说明如下:

STX (通讯起始)

ASCII 模式: ‘:’ 字符。

RTU 模式: 超过 4 个字节的通讯时间 (根据通讯速度不同而自动改变) 的静止时间。

ADR (通讯地址)

合法的通讯地址范围为 1 到 254 之间。

例如对地址为 32 (十六进制为 20) 的伺服进行通讯:

ASCII 模式: ADR= ‘2’, ‘0’ => ‘2’ =32_H, ‘0’ =30_H

RTU 模式: ADR=20_H

CMD (命令指令) 及 DATA (数据)

数据的格式根据命令码而定。常用的命令码如下:

命令码: 03_H, 读取 N 个字 (word), N 最大为 20。

例如: 从地址为 01_H 的伺服读取从起始地址 0200_H 开始的 2 个字。

ASCII 模式:

指令信息:

STX	‘:’
ADR	‘0’
	‘1’
CMD	‘0’
	‘3’
起始数据位置	‘0’
	‘2’
	‘0’
	‘0’
数据数目	‘0’
	‘0’
	‘0’
	‘2’
LRC Check	‘F’
	‘8’
End 1	(0DH)(CR)
End 0	(0AH)(LF)

回应信息:

STX	‘:’
ADR	‘0’
	‘1’
CMD	‘0’
	‘3’
数据数 (以 byte 计算)	‘0’
	‘4’
起始数据地址 0200H 的内容	‘0’
	‘0’
	‘B’
	‘1’
第二笔数据地址 0201H 的内容	‘1’
	‘F’
	‘4’
	‘0’
LRC Check	‘E’
	‘8’
End 1	(0DH)(CR)
End 0	(0AH)(LF)

RTU 模式:

指令信息:

ADR	01H
CMD	03H
起始数据位置	02H (高字节)
	00H (低字节)
数据数 (以 word 计算)	00H
	02H
CRC Check Low	C5H (低字节)
CRC Check High	B3H (高字节)

回应信息:

ADR	01H
CMD	03H
数据数 (以 byte 计算)	04H
起始数据地址	00H (高字节)
0200H 的内容	B1H (低字节)
第二笔数据地址	1FH (高字节)
0201H 的内容	40H (低字节)
CRC Check Low	A3H (低字节)
CRC Check High	D4H (高字节)

指令码: 06_H, 写入 1 个字 (word)例如: 将 100 (0064_H) 写入到局号 01_H 伺服的地址 0200_H。**ASCII 模式:**

指令信息:

STX	':'
ADR	'0'
	'1'
CMD	'0'
	'6'
起始数据地址	'0'
	'2'
	'0'
	'0'
数据内容	'0'
	'0'
	'6'
	'4'
LRC Check	'9'
	'3'
End 1	(0DH)(CR)
End 0	(0AH)(LF)

回应信息:

STX	':'
ADR	'0'
	'1'
CMD	'0'
	'6'
起始数据地址	'0'
	'2'
	'0'
	'0'
数据内容	'0'
	'0'
	'6'
	'4'
LRC Check	'9'
	'3'
End 1	(0DH)(CR)
End 0	(0AH)(LF)

RTU 模式:

指令信息:

ADR	01H
CMD	06H
起始数据地址	02H (高字节)
	00H (低字节)
数据内容	00H (高字节)
	64H (低字节)
CRC Check Low	89H (低字节)
CRC Check High	99H (高字节)

回应信息:

ADR	01H
CMD	06H
起始数据地址	02H (高字节)
	00H (低字节)
数据内容	00H (高字节)
	64H (低字节)
CRC Check Low	89H (低字节)
CRC Check High	99H (高字节)

LRC (ASCII 模式) 和 CRC (RTU 模式) 侦误值的计算:

ASCII 模式的 LRC 计算:

ASCII 模式采用 LRC (Longitudinal Redunancy Check) 侦误值。LRC 侦误值是从 ADR 至最后一笔资料内容之和, 得到之结果以 256 为单位, 去除超出的部分 (例如加总后得到的结果为十六进制的 128_H, 则只取 28_H), 然后计算其补数, 最后得到的结果即为 LRC 侦误值。

例如: 从局号 01_H 伺服驱动器的 0201 地址读取 1 个字 (word)。

STX	‘:’
ADR	‘0’
	‘1’
CMD	‘0’
	‘3’
起始数据地址	‘0’
	‘2’
	‘0’
	‘1’
数据数	‘0’
	‘0’
	‘0’
	‘1’
LRC Check	‘F’
	‘8’
End 1	(0DH)(CR)
End 0	(0AH)(LF)

从 ADR 的数据加至最后一笔数据：

$01_H + 03_H + 02_H + 01_H + 00_H + 01_H = 08_H$ ，对 08_H 取 2 的补数为 $F8_H$ ，所以 LRC 为 ‘F’，‘8’。

RTU 模式的 CRC 计算：

RTU 模式采用 CRC（Cyclical Redundancy Check）侦误值。

CRC 侦误值计算步骤如下：

步骤一：载入一个内容为 $FFFF_H$ 的 16-bit 寄存器，称之为“CRC”寄存器。

步骤二：将指令讯息的第一个位（bit0）与 16-bit CRC 寄存器的低位位（LSB）进行 XOR 运算，并将结果存回 CRC 寄存器；

步骤三：检查 CRC 寄存器的最低位（LSB），若此位为 0，则 CRC 寄存器值右移一位；若此位为 1，则 CRC 寄存器值右移一位后，再与 $A001_H$ 进行 XOR 运算；

步骤四：回到步骤三，直到步骤三已被执行过 8 次，才进行到步骤五；

步骤五：对指令讯息的下一位重复步骤二到步骤四，直到所有位都被这样处理过，此时 CRC 寄存器的内容即是 CRC 侦误值。

说明：计算出 CRC 侦误值之后，在指令讯息中，须先填上 CRC 的低位，再填上 CRC 的高位，请参考以下例子。

例如：从局号为 01_H 伺服的 0101_H 地址读取 2 个字（word）。从 ADR 至资料数的最后一位所计算出的 CRC 寄存器的最后内容为 3794_H ，则其指令讯息如下所示，须注意的是 94_H 在 37_H 的前面传送。

ADR	01_H
CMD	03_H
起始资料地址	01_H (地址高位)
	01_H (地址低位)
资料数 (以 word 计算)	00_H (高位)
	02_H (低位)
CRC 校验低位	94_H (校验低位)
CRC 校验高位	37_H (校验高位)

End1、End0（通讯侦完成）

ASCII 模式:

以 (0D_H) 即字符 ‘\r’ 『carriage return』 及 (0A_H) 即字符为 ‘\n’ 『new line』, 代表通讯结束。

RTU 模式:

超过当前通讯速率下的 4 个字节通讯时间的静止时段表示通讯结束。

范例:

下面以 C 语言产生 CRC 值。此函数需要两个参数:

```
unsigned char * data;
unsigned char length;
此函数将回传 unsigned integer 型态的 CRC 值。
unsigned int crc_chk(unsigned char * data,unsigned char length){
    int i,j;
    unsigned int crc_reg = 0xFFFF;
    while(length- -){
        crc_reg ^=*data++;
        for(j=0;j<8;j++){
            if(crc_reg & 0x01){
                crc_reg=( crc_reg >>1)^0xA001;
            }
            Else
            {
                crc_reg=crc_reg >>1;
            }
        }
    }
    return crc_reg;
}
```

■ **通讯出错**

在通讯过程中, 有可能会发生错误, 常见错误源如下:

- 读写参数时, 数据地址不对;
- 写参数时, 数据超过此参数的最大值或者小于此参数的最小值;
- 通讯受到干扰, 数据传输错误或者校验码错误。

当出现上述前两种通讯错误时, 伺服驱动器运行不受影响, 同时伺服驱动器会反馈回一错误帧。当出现第三种错误时, 传输数据将会被认为无效丢弃, 不返回帧。

错误帧格式如下:

上位机数据帧:

start	从站地址	命令	数据地址、资料等	校验
		命令		

伺服驱动器反馈错误帧:

start	从站地址	响应代码	错误代码	校验
		命令+80 _H		

其中，错误帧响应代码=命令+80_H；

错误代码=00_H：通讯正常；

=01_H：伺服驱动器不能识别所请求的功能；

=02_H：请求中给出的数据地址在伺服驱动器中不存在；

=03_H：请求中给出的数据在伺服驱动器中不允许（超过参数的最大或最小值）；

=04_H：伺服驱动器已经开始执行请求，但不能完成该请求；

例如：伺服驱动器轴号为 03_H，对参数 Pn100 写入数据 06_H，由于参数 Pn100 参数范围 0~6，所以写入数据将不被允许，伺服驱动器将返回一个错误帧，错误代码为 03_H（超过参数的最大或最小值），结构如下：

上位机数据帧：

start	从站地址	命令	数据地址、资料等	校验
	03 _H	06 _H	0002 _H 0006 _H	

伺服驱动器反馈错误帧：

start	从站地址	响应代码	错误代码	校验
	03 _H	86 _H	03 _H	

另外，如果上位机发送的数据帧中的从站地址为 00_H，表示此帧数据是广播数据，伺服驱动器将不返回帧。

6.4 MODBUS 通讯地址

通讯数据地址	含义	说明	操作
十六进制			
0000 _h ~ 03FF _h	参数区	对应参数表中的参数	可读可写
0400 _h ~ 0409 _h	报警信息存储区	10 个历史报警	只读
0410 _h	速度指令零点偏移量		只读
0411 _h	转矩指令零点偏移量		只读
0412 _h	Iu 零点偏移量		只读
0413 _h	Iv 零点偏移量		只读
0420 _h ~ 0437 _h	监控数据		只读
0420 _h	电机转速	单位：1r/min	只读
0422 _h	旋转角（电气角）	单位：1deg	只读
0424 _h	输入指令脉冲速度	单位：1kHz	只读
0426 _h	母线电压	单位：1V	只读
0428 _h	模拟输入速度指令值	单位：1 r/min	只读
042A _h	模拟输入转矩指令百分比	单位：1%	只读
042C _h	内部转矩指令百分比	单位：1%或 0.1A	只读
042E _h	输入信号监视	——	只读
0430 _h	输出信号监视	——	只读

通讯数据地址 十六进制	含义	说明	操作
0432 _h	编码器信号监视	——	只读
0434 _h	输入指令脉冲计数器	单位：1 指令脉冲	只读
0436 _h	反馈脉冲计数器	单位：1 指令脉冲	只读
0438 _h	位置偏移量计数器	单位：1 指令脉冲	只读
043A _h	累计负载	单位：1%	只读
043C _h	转动惯量百分比	单位：1%	只读
043E _h	编码器实际角度	单位：1 指令脉冲	只读
0440 _h	编码器多圈位置	单位：1 圈	只读
044A _h	当前报警		只读
0451 _h	通讯 IO 信号 *1	掉电不保存	可读可写
0452 _h	通讯输出取反	掉电不保存	可读可写
0457 _h	伺服操作状态 *2		只读
045E _h	软件版本号		只读
045F _h	FPGA 版本号		只读
0520 _h	清除历史报警	1：清除历史报警	可读可写
0521 _h	清除当前报警	1：清除当前报警	可读可写
0522 _h	清除总线编码器报警	1：清除总线编码器报警	可读可写
0523 _h	清除总线编码器多圈数据	1：清除总线编码器多圈数据	可读可写
0528 _h	速度 JOG（转速为 P□304 设置）	BIT15:1 JOG 伺服使能 BIT01:1 JOG-（JOG 正转） BIT00:1 JOG+（JOG 反转）	可读可写
0529 _h	位置 JOG（转速为 P□304 设置）	BIT15:1 进入位置点动模式 BIT01:1 JOG- BIT00:1 JOG+	可读可写
0540 _h	恢复出厂值	1：恢复出厂值	可写
0541 _h	复位	1：复位	可写
05F0 _h	当前正在运行数据组号		只读
05F1 _h	即将运行数据组号		只读
05F2 _h	实际位置低 16 位	位置接点电子齿轮后的位置	只读
05F3 _h	实际位置高 16 位		只读
05F4 _h	位置节点方式	0：任务 1：外部	只读
05F5 _h	加速度	10rpm/s/s	可读可写
05F6 _h	减速度	10rpm/s/s	可读可写
05F7 _h	紧急减速度	10rpm/s/s	可读可写
05F8 _h	位置接点电子齿轮分子		可读可写
05F9 _h	位置接点电子齿轮分母		可读可写
05FA _h	找参考点方式		可读可写

通讯数据地址 十六进制	含义	说明	操作
05FB _h	找参考点开关速度	0~6000 rpm	可读可写
05FC _h	离开参考点开关速度	0~6000 rpm	可读可写
05FD _h	示教位置低位		可读可写
05FE _h	示教位置高位		可读可写
数据组 0 参数:			
0600 _h	目标位置低位		可读可写
0601 _h	目标位置高位		可读可写
0602 _h	目标速度	rpm	可读可写
0603 _h	换步属性 *3		可读可写
0604 _h	换步条件 1 数值		可读可写
0605 _h	换步条件 2 数值		可读可写
0606 _h	后续数据组号		可读可写
0607 _h	数据组类型	0: NULL; 1: 绝对; 2: 相对	可读可写
数据组 1 参数:			
0608 _h	目标位置低位		可读可写
0609 _h	目标位置高位		可读可写
060A _h	目标速度	rpm	可读可写
060B _h	换步条件属性		可读可写
060C _h	换步条件 1 数值		可读可写
060D _h	换步条件 2 数值		可读可写
060E _h	后续数据组号		可读可写
060F _h	数据组类型	0: NULL; 1: 绝对; 2: 相对	可读可写
数据组 2 参数:			
0610 _h	目标位置低位		可读可写
0611 _h	目标位置高位		可读可写
0612 _h	目标速度	rpm	可读可写
0613 _h	换步条件属性		可读可写
0614 _h	换步条件 1 数值		可读可写
0615 _h	换步条件 2 数值		可读可写
0616 _h	后续数据组号		可读可写
0617 _h	数据组类型	0: NULL; 1: 绝对; 2: 相对	可读可写
数据组 3 参数:			
0618 _h	目标位置低位		可读可写
0619 _h	目标位置高位		可读可写
061A _h	目标速度	rpm	可读可写
061B _h	换步条件属性		可读可写
061C _h	换步条件 1 数值		可读可写
061D _h	换步条件 2 数值		可读可写

地址说明：

*1、通讯 IO 输入（0451_h）

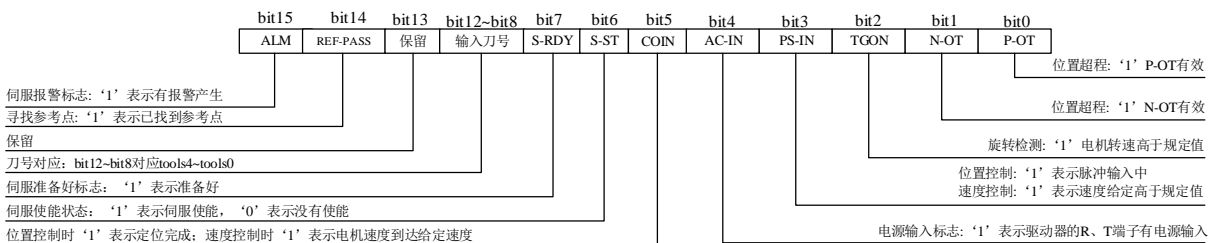
输入信号可通过 MODBUS 通讯的通讯 IO 输入（0451_h）寄存器给定，该寄存器定义如下：

bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
/START-HOME	/POS-STEP	/POS-START	/POS-REF	/POS2	/POS1	/POS0	/G-SEL
bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
/N-CL	/P-CL	/CLR	/ALM-RST	N-OT	P-OT	/P-CON	/SON

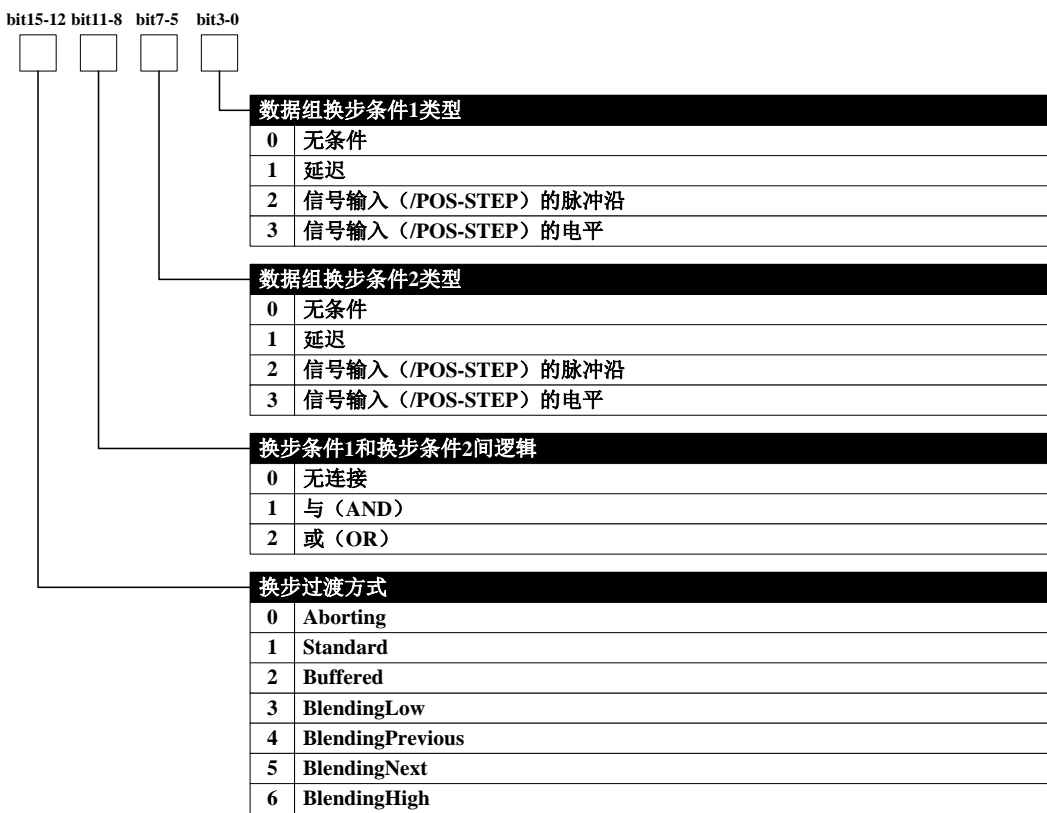
仅在该信号不从 CN3 输入（信号分配参数设置成“无效”）时，该寄存器中信号输入才有效。

例：通过通讯 IO 输入寄存器输入/POS-START，首先需要设置 P□512.1=0，修改通讯 IO 输入（0451_h）寄存器的 bit13 位才有效。

*2、伺服操作状态（0457_h）



*3、换步条件属性



P□503

同速检测信号宽度

设定范围	设定单位	出厂设定	电源重起
0 ~ 100	1r/min	10	不需要

如果电机转速与指令速度之差低于P□503 的设定值，则输出 “/V-CMP” 信号。

■例

P□503=100、指令速度为2000r/min 时，如果电机转速处在1900 ~ 2100r/min之间，则将 “/V-CMP” 置为ON。

电机转速

■补充

“/VCMP” 信号是速度控制时的输出信号。如果是位置控制，则功能自动地变为 “/COIN”，如果是扭矩控制，则自动地变为 “OFF (H 电平)”。