Dotpeenator®

BY SERMATEK

GUIA DE INTEGRAÇÃO · MODBUS TCP

Guia de Registradores Modbus.

Mapa de registradores e exemplos em nível de quadro para conectar as máquinas de marcação por micropercussão Dotpeenator® a CLPs e computadores via Modbus TCP: seleção de arquivo, conteúdo, posição e tamanho dos objetos, controle da marcação e E/S.

PROTOCOLO	PORTA	UNIT ID	CODIFICAÇÃO
Modbus TCP	502	1	UTF-8

Nos nomes de status e identificadores de eventos (por exemplo `Printing`, `PrintFinished`, `OnPrintStart`), a palavra *print* refere-se à marcação na máquina Dotpeenator®. Esses identificadores são mostrados exatamente como o controlador os utiliza.

1 Visão Geral

A máquina de marcação Dotpeenator® se comunica com estações mestre externas (CLPs, computadores supervisores etc.) via protocolo Modbus. A máquina (escravo) atua como servidor Modbus e disponibiliza quatro áreas de dados para leitura e escrita pelo mestre:

ÁREA DE DADOS	TIPO	FAIXA DE ENDEREÇOS	QUANTIDADE	ACESSO
Coils	Bit	0 ~ 8	9	Leitura/Escrita
Discrete Inputs	Bit	0 ~ 11	12	Somente leitura
Input Registers	Palavra de 16 bits	0 ~ 9	10	Somente leitura
Holding Registers	Palavra de 16 bits	0 ~ 65534	65535	Leitura/Escrita

PARÂMETROS DE COMUNICAÇÃO PADRÃO

PARÂMETRO	VALOR
Protocolo	Modbus TCP

PARÂMETRO	VALOR
Porta	502
Endereço do escravo (Unit ID)	1
Codificação de caracteres	UTF-8

CÓDIGOS DE FUNÇÃO

CÓDIGO DE FUNÇÃO	OPERAÇÃO	ÁREA DE DADOS
`0x01`	Ler Coils	Coils
`0x02`	Ler Discrete Inputs	Discrete Inputs
`0x03`	Ler Holding Registers	Holding Registers
`0x04`	Ler Input Registers	Input Registers
`0x05`	Escrever um Coil	Coils
`0x06`	Escrever um Holding Register	Holding Registers
`0x0F`	Escrever múltiplos Coils	Coils
`0x10`	Escrever múltiplos Holding Registers	Holding Registers

2 Modos de Registrador

O controlador suporta dois modos de armazenamento de strings nos Holding Registers, selecionáveis na janela de configurações do módulo Modbus do software do controlador.

MODO	DESCRIÇÃO
Direct (byte único)	Cada byte é armazenado em um registrador usando apenas os 8 bits inferiores; um registrador de valor zero encerra a string
Packed (dois bytes)	Cada dois bytes são compactados em um registrador (primeiro byte nos 8 bits inferiores, segundo nos 8 bits superiores); um registrador de valor zero encerra a string

Ambos os modos usam o mesmo mapa de endereços; muda apenas a forma como as strings são codificadas nos registradores. O padrão é o modo **Direct**, e este documento o utiliza principalmente nas explicações.

3 Mapa de Endereços dos Holding Registers

Os Holding Registers são a área de dados principal, usada para operações de arquivo, controle da marcação e leitura/escrita de dados dos objetos.

FAIXA DE ENDEREÇOS	FINALIDADE	REGISTRADORES
0 ~ 99	Nome do arquivo (sem extensão)	100
100	Reservado	1
101	Comando de controle da marcação	1
102 ~ 999	Reservado	898

FAIXA DE ENDEREÇOS	FINALIDADE	REGISTRADORES
1000 ~ 1099	Área de dados do objeto 0	100
1100 ~ 1199	Área de dados do objeto 1	100
…	…	…
1000+i×100 ~ 1099+i×100	Área de dados do objeto i	100
…	…	…
10900 ~ 10999	Área de dados do objeto 99	100
11000 ~ 65534	Reservado	54535

ESTRUTURA DE CADA ÁREA DE DADOS DE OBJETO (100 REGISTRADORES)

OFFSET	ENDEREÇO (OBJETO I)	FINALIDADE	REGISTR.
+0 ~ +91	1000+i×100 ~ 1091+i×100	Conteúdo do objeto (texto)	92
+92 ~ +93	1092+i×100 ~ 1093+i×100	Posição X (float CDAB)	2
+94 ~ +95	1094+i×100 ~ 1095+i×100	Posição Y (float CDAB)	2
+96 ~ +97	1096+i×100 ~ 1097+i×100	Largura W (float CDAB)	2
+98 ~ +99	1098+i×100 ~ 1099+i×100	Altura H (float CDAB)	2

4 Modo Direct (Byte Único) em Detalhe

4.1 Registradores de Conteúdo do Objeto

> **Endereço:** `1000+i×100 ~ 1091+i×100` (92 registradores por objeto)

No modo Direct, cada byte da string é armazenado nos 8 bits inferiores de um registrador; os 8 bits superiores são sempre zero. Um registrador de valor zero deve seguir o último byte como terminador.

REGRAS DE CODIFICAÇÃO

1. Codifique o conteúdo do objeto em um array de bytes usando UTF-8 (ou a codificação configurada)
2. Armazene cada byte em um registrador (faixa de valores `0x0001 ~ 0x00FF`)
3. Acrescente um registrador de valor zero após o último byte como terminador
4. Preencha os registradores restantes com 0

EXEMPLO: CONTEÚDO DO OBJETO 0 É "ABC"

Bytes UTF-8: `0x41 0x42 0x43`

ENDEREÇO DO REGISTRADOR	VALOR (HEX)	SIGNIFICADO
1000	0x0041	'A'
1001	0x0042	'B'
1002	0x0043	'C'
1003	0x0000	Terminador

ENDEREÇO DO REGISTRADOR	VALOR (HEX)	SIGNIFICADO
1004 ~ 1091	0x0000	Não usado

Exemplo de quadro Modbus — escrever "ABC" no objeto 0 (FC 0x10)

REQUISIÇÃO

00 01	00 00	00 0F	01	10	03 E8	00 04	08	00 41	00 42	00 43	00 00
TID	PID	Len	Unit	FC	Addr	Qty	BC	Reg0	Reg1	Reg2	Reg3
	0	15	1	0x10	1000	4	8	'A'	'B'	'C'	Fim

RESPOSTA

00 01	00 00	00 06	01	10	03 E8	00 04
TID	PID	Len	Unit	FC	Addr	Qty
	0	6	1	0x10	1000	4

EXEMPLO: CONTEÚDO DO OBJETO 1 É "HELLO"

Bytes UTF-8: `0x48 0x65 0x6C 0x6C 0x6F` . Endereço base do objeto 1 = 1000 + 1×100 = **1100**.

ENDEREÇO DO REGISTRADOR	VALOR (HEX)	SIGNIFICADO
1100	0x0048	'H'
1101	0x0065	'e'
1102	0x006C	'l'
1103	0x006C	'l'
1104	0x006F	'o'
1105	0x0000	Terminador
1106 ~ 1191	0x0000	Não usado

Exemplo de quadro Modbus — escrever "Hello" no objeto 1 (FC 0x10)

REQUISIÇÃO

00 02	00 00	00 13	01	10	04 4C	00 06	0C	00 48	00 65	00 6C	00 6C	00 6F	00 00
TID	PID	Len	Unit	FC	Addr	Qty	BC	Reg0	Reg1	Reg2	Reg3	Reg4	Reg5
	0	19	1	0x10	1100	6	12	'H'	'e'	'l'	'l'	'o'	Fim

RESPOSTA

00 02	00 00	00 06	01	10	04 4C	00 06
TID	PID	Len	Unit	FC	Addr	Qty
	0	6	1	0x10	1100	6

EXEMPLO COM CARACTERES MULTIBYTE: CONTEÚDO DO OBJETO 0 É "ÇELIK"

Bytes UTF-8: `0xC3 0x87 0x65 0x6C 0x69 0x6B` (6 bytes — "Ç" ocupa 2 bytes)

ENDEREÇO DO REGISTRADOR	VALOR (HEX)	SIGNIFICADO
1000	0x00C3	Byte UTF-8 1 de 'Ç'
1001	0x0087	Byte UTF-8 2 de 'Ç'
1002	0x0065	'e'
1003	0x006C	'l'
1004	0x0069	'i'