



DMC-2x00 数字运动控制器

使用说明书

北京宝伦数控技术有限公司

POWERLAND Technologies Co.

公司总部： 地址：北京市海淀区上地三街9号嘉华大厦F座10层 邮编：100085
电话：010-82840851, 82840855, 82890324, 82890854 传真：010-82841734

Internet: www.power-land.com

E-mail : sales@power-land.com

上海办事处：

广州办事处：

电话：021-62511327, 62512536

电话：020-38628383, 13929579393

目 录

序-----	8
第一篇 用户使用手册-----	9
第一章 概述-----	9
1 说明-----	9
2 电机接口类型-----	9
3 放大器种类-----	10
4 DMC-2X00 功能框图-----	10
第二章 控制系统建立-----	11
1 系统构成-----	11
2 安装 DMC-2X00-----	12
3 设计编程入门-----	21
第三章 硬件接口说明与连接-----	26
1 概述-----	26
2 输入接口-----	26
3 扩展 I/O 接口-----	28
4 控制器对数据的响应-----	30
5 查询控制器-----	30
第四章 通信-----	31
1 说明-----	31
2 RS-232 口-----	31
3 Ethernet 配置 (DMC-2100/2200) -----	33
4 数据纪录-----	36
5 控制器对命令的响应-----	41
6 由控制器产生的非请求信息-----	41
7 GALIL 软件工具和库文件-----	41
第五章 命令基础-----	42
1 说明-----	42
2 命令语法---ASCII 码-----	42
3 命令语法---二进制码-----	43
第六章 运动编程-----	45
1 概述-----	45
2 独立轴定位控制-----	46

3	独立 JOG 运动-----	48
4	直线插补运动-----	50
5	直线和圆弧插补方式-----	54
6	电子齿轮同步功能-----	58
7	电子凸轮-----	60
8	轮廓方式-----	64
9	虚拟轴功能-----	68
10	步进机工作方式-----	69
11	双闭环（辅助编码器）-----	71
12	运动平滑（S 曲线加减速）-----	73
13	原点返回功能-----	74
14	高速位置捕获（位置锁存功能）-----	77
第七章 应用编程-----		77
1	概述-----	77
2	程序编辑-----	78
3	程序格式-----	79
4	执行程序---多任务-----	81
5	调试程序-----	82
6	程序流程命令-----	84
7	数学及函数表达式-----	95
8	变量-----	96
9	操作数-----	98
10	数组-----	99
11	数据输入（数学和字符串）-----	101
12	数据输出（数学和字符串）-----	104
13	硬件 I/O-----	108
14	DMC-2X00 控制器扩展 I/O-----	111
15	应用举例-----	113
第八章 硬件保护和软件保护-----		118
1	说明-----	118
2	硬件保护-----	118
3	软件保护-----	119
第九章 故障处理-----		121
1	概述-----	121
2	安装-----	121
3	通信-----	122
4	稳定性-----	122
5	操作-----	122

第二篇 命令参考手册-----123**第一章 概述-----123**

1	控制器注解-----	123
2	伺服电机和步进电机注解-----	123
3	命令说明-----	123
4	二进制命令-----	125
5	控制软件加速运行-----	127
6	条件启动-----	128
7	指令集-----	129

第二章 指令详解-----134

1	AB-----	134
2	AC-----	135
3	AD-----	135
4	AF-----	136
5	AI-----	137
6	AL-----	138
7	AM-----	138
8	AO-----	139
9	AP-----	140
10	AR-----	140
11	AS-----	141
12	AT-----	142
13	AV-----	142
14	BA-----	143
15	BB-----	144
16	BC-----	144
17	BD-----	145
18	BG-----	145
19	BI-----	146
20	BL-----	147
21	BM-----	147
22	BN-----	148
23	BO-----	149
24	BP-----	149
25	BS-----	150
26	BV-----	151
27	BZ-----	151
28	CA-----	152
29	CB-----	153
30	CC-----	153
31	CD-----	154

32	CE-----	154
33	CF-----	155
34	CI-----	155
35	CM-----	156
36	CN-----	156
37	CO-----	157
38	CR-----	158
39	CS-----	159
40	CW-----	159
41	DA-----	160
42	DC-----	161
43	DE-----	161
44	DL-----	162
45	DM-----	163
46	DP-----	163
47	DR-----	164
48	DT-----	164
49	DV-----	165
50	EA-----	166
51	EB-----	166
52	EC-----	167
53	ED-----	167
54	EG-----	168
55	EI-----	169
56	ELSE-----	170
57	EM-----	170
58	EN-----	171
59	ENDIF-----	172
60	EO-----	172
61	EP-----	173
62	EQ-----	173
63	ER-----	174
64	ES-----	175
65	ET-----	175
66	FA-----	176
67	FE-----	176
68	FI-----	177
69	FL-----	178
70	FV-----	179
71	GA-----	179
72	GM-----	180
73	GR-----	181
74	HM-----	181
75	HS-----	182

76	HX-----	182
77	IA-----	183
78	IF-----	184
79	IH-----	184
80	II-----	186
81	IL-----	187
82	IN-----	187
83	IP-----	188
84	IT-----	189
85	JG-----	189
86	JP-----	190
87	JS-----	191
88	KD-----	191
89	KI-----	192
90	KP-----	192
91	KS-----	193
92	LA-----	193
93	LE-----	194
94	-LF*-----	195
95	LI-----	195
96	LL-----	196
97	LM-----	196
98	-LR*-----	197
99	LS-----	198
100	LV-----	198
101	LZ-----	199
102	MB-----	200
103	MC-----	201
104	MF-----	202
105	MG-----	202
106	MO-----	203
107	MR-----	204
108	MT-----	204
109	NB-----	205
110	NF-----	206
111	NO-----	206
112	NZ-----	206
113	OB-----	207
114	OC-----	207
115	OE-----	208
116	OF-----	209
117	OP-----	209
118	PA-----	210
119	PF-----	211

120	PL-----	212
121	PR-----	213
122	QD-----	213
123	QR-----	214
124	QU-----	214
125	QZ-----	215
126	RA-----	215
127	RC-----	216
128	RD-----	216
129	RE-----	218
130	RI-----	218
131	RL-----	219
132	RP-----	220
133	RS-----	220
134	〈control〉 R 〈control〉 S-----	221
135	〈control〉 R 〈control〉 V-----	221
136	SA-----	221
137	SB-----	222
138	SC-----	222
139	SH-----	223
140	SP-----	223
141	ST-----	224
142	TB-----	225
143	TC-----	225
144	TD-----	227
145	TE-----	227
146	TH-----	228
147	TI-----	229
148	TIME*-----	229
149	TL-----	230
150	TM-----	230
151	TN-----	231
152	TP-----	232
153	TR-----	232
154	TS-----	233
155	TT-----	233
156	TV-----	234
157	TW-----	234
158	TZ-----	235
159	UI-----	236
160	UL-----	236
161	VA-----	237
162	VD-----	238
163	VE-----	238

164 VF-----	239
165 VM-----	240
166 VP-----	241
167 VR-----	242
168 VS-----	242
169 VT-----	243
170 WC-----	244
171 WH-----	244
172 WT-----	245
173 XQ-----	245
174 ZS-----	246
 第三章 附录-----	 247
1 典型应用领域-----	247
2 相关链接产品-----	247
3 其它产品-----	247

序

为了使广大用户正确使用、快速理解 DMC-2x00 数字运动控制器，北京宝伦数控技术有限公司以美国 GALIL 公司 DMC-2x00 用户手册英文版及宝伦公司技术人员多年使用实践积累的经验，编写此中文版《DMC-2x00 数字运动控制器使用说明书》。此书共分二部分，第一部分为用户使用手册，第二部分为软件编程用命令参考手册，希望能满足用户的迫切愿望。若本书能够对广大用户快速理解、正确使用 DMC-2x00 数字运动控制器有所帮助，中宝伦公司员工及编者本人将不甚感激。

美国 GALIL 公司是以专业从事数字运动控制器开发、生产、销售而在世界久负盛名的公司，GALIL 公司总裁 Jacob TAL 先生被公认为学者、专家，历经 20 年发展，已有 20 多个系列的数字运动控制产品问世，截止 2008 年，该公司有 50 万台套数字运动控制器产品在全世界广泛应用，可靠运行。

GALIL 公司数字运动控制器种类齐全，涉及 PCI 总线、ISA 总线、VME 总线、CPCI 总线、PC/104 总线、Ethernet 总线、USB 总线、RS232/422 等多种形式，每种形式又分为 1 1/2 轴、1~8 轴供用户选择。任何一款控制器均可控制伺服电机或步进电机或液压马达，也可以是其中的任意组合。每种结构形式的控制器均采用 32 位处理器、Flash 存储器（用于存储用户程序、参数、变量、数组）、RAM、FIFO，因此，控制器运行可独立进行，不占用主机（上位机）资源。除基本 I/O、限位、原点开关、急停、报警、复位等输入/输出接口之外，也有相应的扩展 I/O，供用户选择。对于高档产品，还有第二编码器接口、模拟量输入接口，这对于高精度、复杂控制要求的机电一体化设备极其有用。

在应用编程方面，GALIL 公司提供了功能强大的 API，用户可以在 DOS、Windows3.x, 95, 98, 2000, ME、NT、Linux、CE、QNX 环境下，用 VB、VC、C/C++、Labview、Delphi 等工具进行应用程序开发；特别与众不同是 GALIL 公司还提供了简单明了、极易使用的 2 字符命令集，对于初次使用者来说，犹如“ABC”般易学易懂，而且对于许多应用来讲，在集成化的智能终端环境下，编辑、下载、调试、烧录程序可一气呵成。关于这些独特的优点，在后续的章节中将做更为详细的说明。

另外一个需要特别强调的鲜明特点，即对于 GALIL 公司 20 多个系列 DMC 数字运动控制器来说，编程方法、功能命令除硬件结构有所不同之外，完全兼容，也就是说，虽然本手册针对 DMC-2x00 控制器，但除通信形式不同之外，对于其它系列的控制器仍然有宝贵的参考价值。

由于本人编译水平有限，本手册中的错误在所难免，恳请广大读者、专家指正赐教。期盼本手册对于热衷于从事数字运动控制、自动化精密机械设计制造的同仁们有所帮助，让我们携手共创中国制造业自动化之美好未来！

参加翻译的人员还有北京中宝伦自动化技术有限公司的梅华、孙杰、景志超、刘敬春等同志，校对工作由陈玉明同志完成，同时，在编译过程中，得到了北京邮电大学自动化学院副院长杨军教授的大力支持。在此，一并表示诚挚的谢意！

编译：景琦

第一篇 用户使用手册

第一章 概述

1. 说明:

DMC-2x00 系列是 GALIL 公司最高性能独立型控制器。该控制器具有许多强大功能，如高速通信、非易失程序存储器、高速编码器反馈接收、高抗干扰性（EMI）等。

每台 DMC-2x00 提供两种通信通道：高速 RS-232（2 路可达 115K 波特率）和 12Mb/s 的 USB（DMC-2000）或 10 BaseT Ethernet（DMC-2100）或 100BaseT Ethernet（DMC-2200），且提供 4M Flash EEPROM 非易失性存储器，用于存储应用程序、参数、变量、数组和控制软件。因此，很容易在现场对控制软件进行升级。

DMC-2x00 可以控制 8 个轴。DMC-2x10, 2x20, 2x30, 2x40 是 1~4 轴控制器，而 DMC2x50, 2x60, 2x70, 2x80 是 5~8 轴控制器。

DMC-2x00 专为解决复杂运动难题而设计，能够用于涉及 JOG、PTP 定位、多轴联动、矢量定位、电子齿轮同步、电子凸轮、多任务、轮廓运动等。控制器通过具有轨迹平滑处理的可编程加、减速可大大减小运动冲击。为了复杂轮廓平滑跟踪，DMC-2x00 还提供无限直线、圆弧线段的矢量进给。该控制器也具有多主、多从电子齿轮及龙门同步控制方式。

为了与外部事件同步，DMC-2x00 提供了通用 I/O，8 路光电隔离数字输入（DMC-2x50~DMC-2x80 为 16 路）、8 路数字输出（DMC-2x50~DMC-2x80 为 16 路），8 路模拟输入用于手操杆、传感器、压力传感器接口。此外，还有 64 点 I/O 扩展接口。如果不使用第二编码器，可以获得更多的 I/O（2 路/轴），为正、负向限位、急停、原点开关、提供专用光电隔离输入接口。

DMC-2x00 可以用二进制或 ASCII 码格式传送命令，并备有许多可选择工具软件，如：用于自动调整的 WSDK 软件、用于轨迹观察的 AUTOCAD 文档转换软件等等，且用户能够用 VB、VC、C/C++ 等进行程序开发。驱动库函数可以适用于 DOS、Linux、Windows3.x, 95, 98, 2000, ME 当前所有版本，NT、CE、QNX 等。

2. 电机接口类型:

2.1 具有±10V 指令信号的标准伺服电机驱动器

由于 DMC-2x00 使用 16 位电机指令输出 DAC 及具有速度/加速度前馈、凹陷滤波、低通滤波和积分限制的高级 PID 滤波器，因此具有超精密控制特性。

出厂时将控制器配置成标准伺服电机驱动器工作方式。在此配置下，控制器输出±10V 模拟指令信号，与伺服放大器相连接。此连接方式在第二章中描述。

2.2 正弦波换向型无刷伺服电机驱动器

鉴于国内目前此种应用较少，因此，这里不做详细说明，若有用户对此感兴趣，请登录本公司网站：www.power-land.com 下载相关资料或与北京中宝伦公司联系。

2.3 具有脉冲+方向控制的步进电机驱动器

DMC-2x00 能够用于控制步进电机。在这种方式中，控制器提供脉冲、方向两路信号与步进电机驱动器相连接。对于步进电机工作方式来说，控制器不需要编码器反馈，而以开环方式工作。具体连接将在第二章中解释。

2.4 与液压马达相连接

DMC-2x00 控制器能够用于控制液压马达，其指令输出信号形式与标准伺服电机相同，只是需要注意控制参数匹配。

3. 放大器种类：

放大器应该与电机相匹配，可以是线性或 PWM 型。放大器具有电流反馈、电压反馈或速度反馈。

3. 1 电流型放大器

电流型放大器接受 $\pm 10V$ 模拟指令信号。按照 $+10V$ 指令产生所需最大电流原则来设置放大器增益。例如，如果电机峰值电流为 $10A$ ，放大器增益就应该是 $1A/V$ 。

3. 2 速度型放大器

对于速度型放大器， $10V$ 指令信号应该使电机以所需最大速度运转。

3. 3 步进电机放大器

步进电机放大器接受指令脉冲+方向信号。

需要说明的是：目前市场上大多数数字伺服将位置环、速度环、电流环集中一体，也可以设置成位置控制方式，接收指令脉冲+方向控制信号，其工作方式类似于步进电机放大器，但切切注意，此工作方式时，数字伺服侧可接受的指令脉冲频率一般只有 $200KHz \sim 500KHz$ ，因此，对于高速、高精密控制需求，建议采用标准伺服电机工作方式（即将伺服放大器设置为速度型），位置控制运算由 DMC-2x00 控制器来完成，此种设置，控制器可接收的反馈编码器速率可达 $12MHz$ ，指令通道亦不会受到限制。

4. DMC-2x00 功能框图：

DMC-2x00 功能框图如图 1.1 所示：

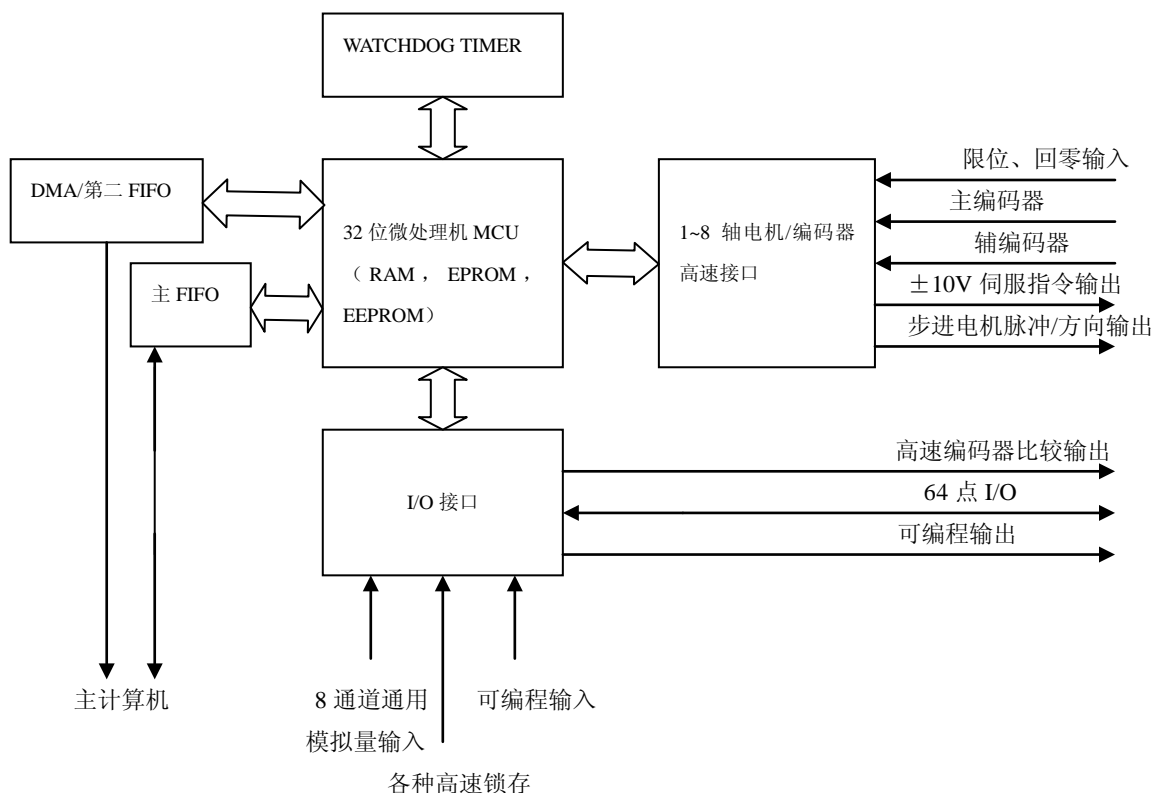


图 1.1

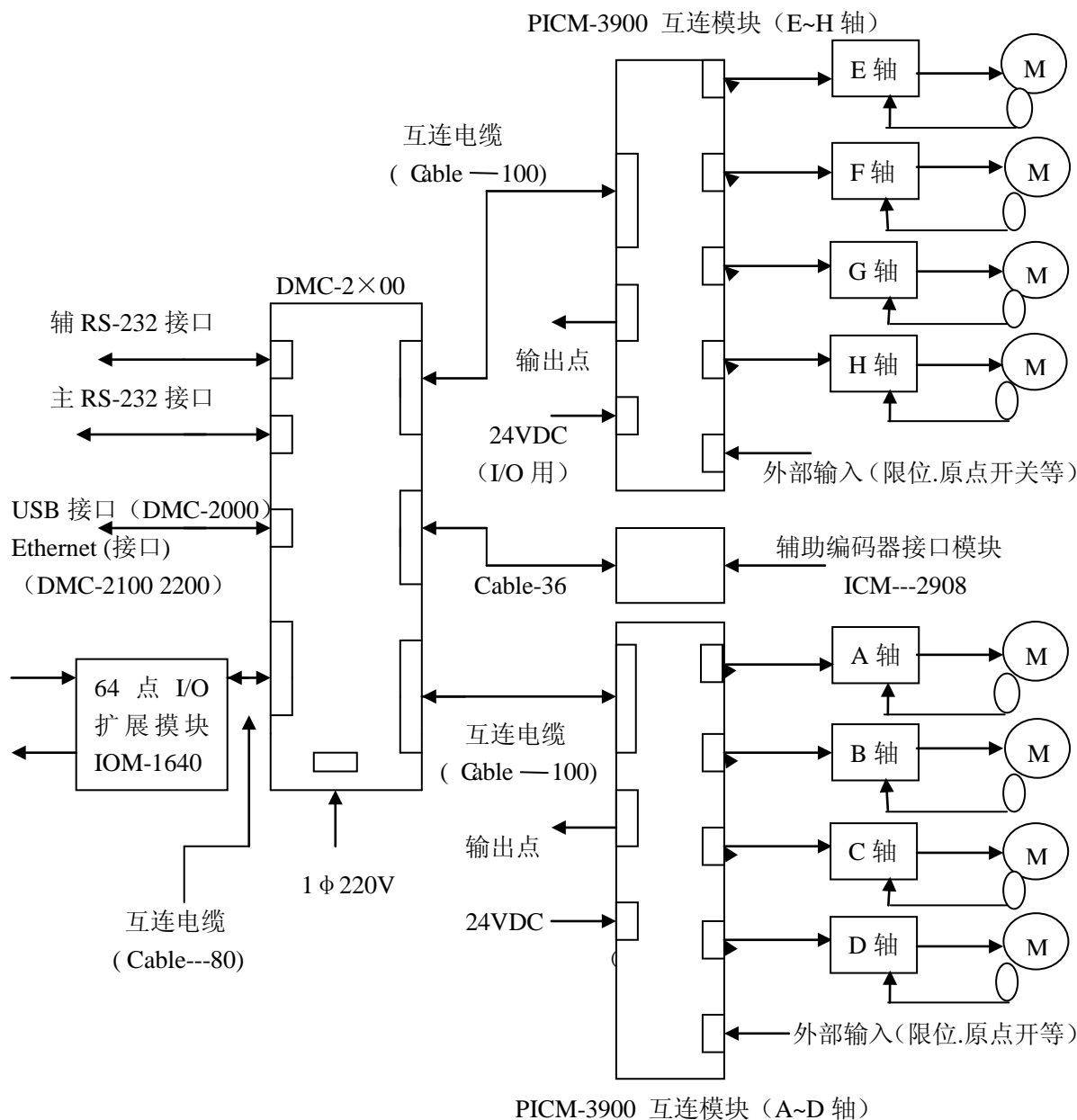
图中各功能部件的说明将在后续章节中进行，这里只对看门狗定时器加以说明。

DMC-2x00 提供内部看门狗定时器，以确保微处理器工作在正常状态。定时器使放大器使能输出(AMPEN)翻转，在 DMC-2x00 出现严重故障时，用 AMPEN 使放大器输出关断。AMPEN 通常为高电平，在上电期间，如果微处理器暂停工作，AMPEN 输出就会变低，报警灯也在此阶段点亮。此时可用复位操作使 DMC-2x00 恢复工作。若您的 DMC-2x00 出现工作异常，请及时与中宝伦公司取得联系。

第二章 控制系统建立

1. 控制系统构成：

控制系统构成如图 2.1 所示：



要构成一套完整的多轴运动控制系统，建议用户按照如下步骤进行配置：

第一步：选择合适的 DMC 控制器

- A. 控制轴数
- B. 总线形式：PCI 总线、CPCI 总线、ISA 总线、PC/104 总线、VME 总线、Ethernet 或 USB 总线、RS232/422 等。
- C. 主要根据您的使用要求、安装形式而定。带有 Ethernet 、USB、RS232/422 接口的均为独立型控制器。

第二步：选择互联模块并确认互联电缆（CABLE-100-xM）的长度。

互联模块的型号为 PICM-3900-OPTO；互联电缆的长度一般提供 1（M）米、2(M)米及 4(M)米共 3 种规格，型号为：Cable-100-1/2/4M。

第三步：确认 I/O 控制点数，以决定是否需要选择 IOM-1964 扩展模块及其互联电缆（CABLE-80-xM）。

第四步：根据控制精度要求确认是否需要第二编码器反馈模块 ICM-2908 及其互联电缆（CABLE-36-xM）。

第五步：根据负载转矩、惯量、加速功率、加工速度及精度选择电机种类（伺服电机或步进电机）及规格大小。对于负载惯量远远大于电机惯量的应用，为确保控制系统的稳定性及良好的动态响应性，建议用户选配合适的减速器与之配套，推荐优选高精密行星齿轮减速器以确保传动精度。

2、安装 DMC-2x00:

第一步：确定电机配置

在建立控制系统之前，用户必须确定所需要的电机配置。DMC-2x00 能够控制标准伺服电机，正弦波换向无刷电机，步进电机或其中的任意组合。其它种类的执行机构（如直线电机、直接驱动电机（DD Motor）、液压伺服马达等）也可连接使用。下面是有关电机配置方面的说明。

A. 标准伺服电机放大器工作方式

出厂时，已将 DMC-2x00 设置成标准伺服电机放大器工作方式，提供±10V 模拟指令输出信号。因此，无需再做硬、软件配置。

B. 步进电机放大器工作方式：

要想将 DMC-2x00 配置成步进电机工作方式，在硬件上对每个轴需要一个用于设定步进电机的短路棒，而且在软件配置方面必须用 MT 命令将所对应的轴设置成步进电机方式。有关详细内容将在后面讨论。

第二步：DMC-2x00 短路棒设定

A. 主复位和升级短路棒：

主板上 JP1 含有两个短路棒、MRST 及 UPGRD。MRST 用于主复位，当插上 MRST 短路棒时，对控制器上电或使复位输入信号变低，控制器就执行主复位操作。也能够用控制器外侧的 DIP 开关设置 MRST。无论何时对控制器进行主复位，都将擦除存储在 EEPROM 中的所有程序、数组、变量和运动控制参数。

用 UPGRD 短路棒能够对控制器的控制软件无条件升级。在控制器正常工作时，对软件升级不需要此短路棒，在 EEPROM 有问题的情况下，才需要此短路棒。一般来说，EEPROM 不会毁坏，如果在控制软件升级期间出现电源故障，有可能使 EEPROM 出问题。若 EEPROM 出现毁坏，您的控制器就不会正常工作。在此情况下，插上 UPGRD 短路棒，使用 GALIL 智能终端（Smart Terminal）中的更新软件功能（Update Firmware）重新装入系统控制软件。

B. 光电隔离短路棒：

GALIL 公司 DMC 控制器高档型，均对数字输入、限位、原点开关进行光电隔离。如果您不需要隔离电源，就可以用来自 PC 的内部+5V 电源对光电隔离供电。此时需要插入主板上的 JP3 短路棒。

注意：中宝伦公司设计的互联模块 PICM-3900-OPTO 对 DMC 控制器的所有输入、输出信号均采用光电隔离处理，因此，无需进行任何设定即可与 GALIL 公司任何一款 DMC 控制器相连接。

C. 步进电机短路棒：

要想使某一轴以步进电机方式工作，必须插入与之相对应的步进方式（SM）短路棒。此短路棒位于主板上，对于 DMC-2x00 来说，标号为 JP5 和 JP7。各个短路棒分别以 SMA~SMH 来标注，当插入短路棒时，即将所对应的轴 A~H 设置成了“步进方式”。请注意，要插、拔这些短路棒，必须取下 DMC-2x00 主板上的子板。

D. 电机关断短路棒：

可以用控制器上的硬件短路棒来选择上电时的电机状态。插入 MO 位置上的短路棒，就会使控制器在“电机关断”状态下上电。为了使电机使能，就需要执行 SH 命令。不插入短路棒，电源一上电，控制器就会立即使电机使能。要关断电机，就需要执行 MO 命令。

MO 短路棒位于步进电机短路棒（SM）的旁边。对于某些 DMC 控制器，无 MO 短路棒。

E. DMC-2000 通信短路棒：

主、付串行通信口通常均为 RS-232 方式，DMC-2000 子板上的短路棒 JP3 和 JP4 可以将 DMC-2000 配置成 RS-422 方式。用户在购买时，作为选择功能需要指明，或者由用户重新配置，相关说明请咨询 GALIL 公司。也能提供其它串行通信协议，如 RS-485，此项作为特殊选择，请咨询 GALIL 公司。

F. DMC-2100/DMC-2200 通信短路棒：

主、付串行通信口通常均为 RS-232 方式，用 DMC2100/2200 子板上的短路棒 JP4 和 JP5 可以将其配置成 RS-422 方式。用户在购买时，作为选择功能需要指明，或者由用户重新配置，相关说明请咨询 GALIL 公司。也能提供其它串行通信协议，如 RS-485，此项作为特殊选择，请咨询 GALIL 公司。

第 3 步：DIP 开关配置：

控制器金属壳外侧有一组 5 位 DIP 开关。当控制器上电或复位时，读取 DIP 开关的状态。

A. 配置 DMC-2000 上的 DIP 开关：

开关 1—主复位

此开关为 ON 时，PC 机一上电，控制器即执行主复位操作。无论何时对控制器进行主复位，都会擦除存储在 EEPROM 中的所有程序、运动控制参数。在正常操作期间，此开关应为 OFF。

开关 2—XON/XOFF

此开关为 ON 时，将使主串口的软件握手有效。

开关 3—硬件握手方式

此开关为 ON 时，将使主串口的硬件握手有效。

开关 4~6—主串口波特率设定，如下表所示：

9600	19.2	3800	波特率
ON	ON	OFF	1200

ON	OFF	OFF	9600
OFF	ON	OFF	19200
OFF	OFF	ON	38400
OFF	ON	ON	115200

开关 10—USB 设定

此开关为 ON 时，控制器将使用 USB 口作为通信的缺省口。当为 OFF 时，控制器将使用 RS-232 口作为通信的缺省口。对控制软件更新时，控制器将发送响应（一个冒号）给设定的缺省口。如果此口与用来下载控制软件的通信口不相同，GALIL 软件就会将操作控制返回给用户。在此情况下，就不得不重新启动软件运行。

B. 配置 DMC-2100 上的 DIP 开关：

开关 1—主复位

此开关为 ON 时，PC 机一上电，控制器就会执行主复位操作。无论何时对控制器进行主复位，都会擦除存储在 EEPROM 中的所有程序和运动控制参数。在正常工作期间，此开关为 OFF。

开关 2—XON/XOFF

此开关为 ON 时，将使主串口的软件握手有效。

开关 3—硬件握手方式

此开关为 ON 时，将使主串口的硬件握手有效。

C. 配置 DMC-2200 上的 DIP 开关：

开关 1—主复位

此开关为 ON 时，PC 机一上电，控制器即执行主复位操作。无论何时对控制器进行主复位，都会擦除存储在 EEPROM 中的所有程序、运动控制参数。在正常操作期间，此开关应为 OFF。

开关 2—XON/XOFF

此开关为 ON 时，将使主串口的软件握手有效。

开关 3—硬件握手方式

此开关为 ON 时，将使主串口的硬件握手有效。

开关 4~6—主串口波特率设定，如下表所示：

9600	19.2	3800	波特率
ON	ON	OFF	1200
ON	OFF	OFF	9600
OFF	ON	OFF	19200
OFF	OFF	ON	38400
OFF	ON	ON	115200

开关 7—选择

当此开关为 OFF 时，控制器将使用自动交互功能来设置 Ethernet 连接速度。当此 DIP 开关为 ON 时，控制器缺省为 10Base T。

开关 8—Ethernet

此开关为 ON 时，控制器将 Ethernet 口用作缺省通信口传送信息。当为 OFF 时，控制器将 RS-232 用作缺省通信口。在更新控制软件时，控制器会发送响应（冒号“:”）给缺省口。

如果下载控制软件所使用的通信口与缺省通信口不相同，GALIL 软件就会将操作控制返回给用户。

第 4 步 安装通信软件:

在对计算机上电后，您应该安装 GALIL 软件，使控制器和 PC 机之间能够通信。

A. 使用 DOS:

用 GALIL 软件 CD-ROM 盘，进入目录 DMCDOS，在 DOS 提示符处键入“INSTALL”，并按提示操作。

B. 使用 Windows3.x (16 bit 版):

用 GALIL 软件 CD-ROM 盘，进入目录 DMCWIN16，在命令提示处运行 DMCWIN16.exe，并按提示操作。

C. 使用 Windows95, 98, NT, ME, 2000 (32 bit 版):

插入 GALIL 软件 CD-ROM 盘时，GALIL 软件 CD-ROM 就会自动开始安装过程。将 GALIL CD-ROM 安装到您的计算机之后，您就能很容易地安装所需要的其它软件工具。要想安装基本通信软件，就运行 GALIL 软件 CD-ROM，并选择“DMCWIN32—Windows 下的工具软件和编程用库文件。此时，就会将 GALIL 集成终端工具软件安装到您的计算机上，用此在计算机与控制器之间建立通信。

注：在最新版的 GALIL 软件 CD-ROM 中，GALIL 集成化智能终端工具软件 (Smart Terminal) 作为一个独立的文档提供给用户，因此，可以单独安装。

第 5 步: 连接 AC 电源到控制器:

通电前，将 100pin 电缆连接到 DMC-2x00 与 PICM-3900 互联模块之间。DMC-2x00 需要单相 50/60Hz, 90~260VAC 电源。

警告:

在 DMC-2x00 相匹配的放大器和伺服电机中有高电压、电流、温度及旋转体，应用连接时务必小心。只有合格人员方可进行安装、配置、操作此装置，当通电后，切勿打开控制器外壳。

通电后，绿色电源指示灯应该点亮。

第 6 步: 用 GALIL 软件建立通信:

A. 由主串口通信:

用 RS-232 电缆将 DMC-2x00 主串口与您的计算机相连接。

a. 使用 DOS 版 GALIL 软件 (只对 DMC-2000)

要与 DMC-2000 通信，在提示行键入 TALKZDMC，一旦您建立了通信，终端显示应出现一个冒号 (:)。如果未接收到冒号，就按回车键。如果没有返回冒号提示，很大可能性是由于不正确的串口设置。用户必须确保在与控制器进行通信时，设置正确的通信口和波特率。请注意：为了与 GALIL 软件进行通信，控制器上的串口必须设置为硬件握手方式。用户也必须确保使用合适的串行通信电缆。串行通信电缆的引脚定义参见附录。

b. 使用 Windows 版 GALIL 软件

为了在 Windows 环境下与 GALIL 控制器进行通信，必须在 Windows Registry (注册表) 中注册控制器。要注册控制器，您必须指定控制器的型号、通信参数及其它相关资料。通过 GALIL 软件进行注册，如 Smart Terminal (或 D TERM 或 DMCTERM) 和 WSDK (用 DMCWIN 来安装 DTERM, 安装为控制按钮“GALIL Terminal”)。用 WSDK, 在 FILE 菜单下进行注册，用 DTERM 或 Smart Term 程序，从 REGISTRY 菜单进行注册。注册窗口配备有按钮 ADD (添加), Change (更改) 或 Delete (删除) 控制器，点击其中任何按钮都会弹出 Set Registry Information (设置注册信息) 窗口。用 Add 按钮来添加一个新控制器入口到 Registry (注册表)。此时需要您提供并输入 GALIL 控制器类别型号，如果您

更改已存在的控制器，此区域也会有入口。点击靠右的下拉箭头就会展现出可用的控制器型号的菜单。然后，您就可以选择串行或 Ethernet 连接。请记住，由 USB 连接的 DMC-2000 是即插即用型，应该自动将 DMC-2000 添加到注册表中。注册信息会显示出缺省通信口 1，缺省速度为 9600。如果需要，就应该更改此信息，以便与计算机通信口和通过控制器 DIP 开关所设定的波特率相一致。注册入口也显示定时器超时和延迟信息。这些属高级参数，应该由高级用户来更改（详细说明请参见软件文档）。一旦您为控制器设置了合适的注册信息，就选择 OK 并关闭注册窗口。至此，您就能够与 DMC-2x00 进行通信了。一旦您选择了入口，点击 OK 按钮即可。如果软件成功地建立起与控制器通信，就会在显示屏顶部显示注册入口。

如果您与控制器通信不正确，程序会暂停 3~15 秒，显示屏顶部会显示信息：“Status: not connected with GALIL motion controller”并出现如下报警提示：“STOP—Unable to establish communication with the GALIL controller. A time-out occurred while waiting for a response from the GALIL Controller。若出现此信息，您必须点击 OK。在此情况下，其原因大多是由于串行通信口不正确的设定。当想与控制器通信时，您必须设置正确的通信口和波特率。请注意：用 GALIL 软件必须将控制器上的串口设置成“硬件握手方式”，同时确保使用合适的串行通信电缆。串行通信电缆的引脚定义参见附录。

一旦您建立通信，点击“Smart Term”菜单，您就会接收到一个冒号（:）提示。与控制器通信在后续章节中描述。

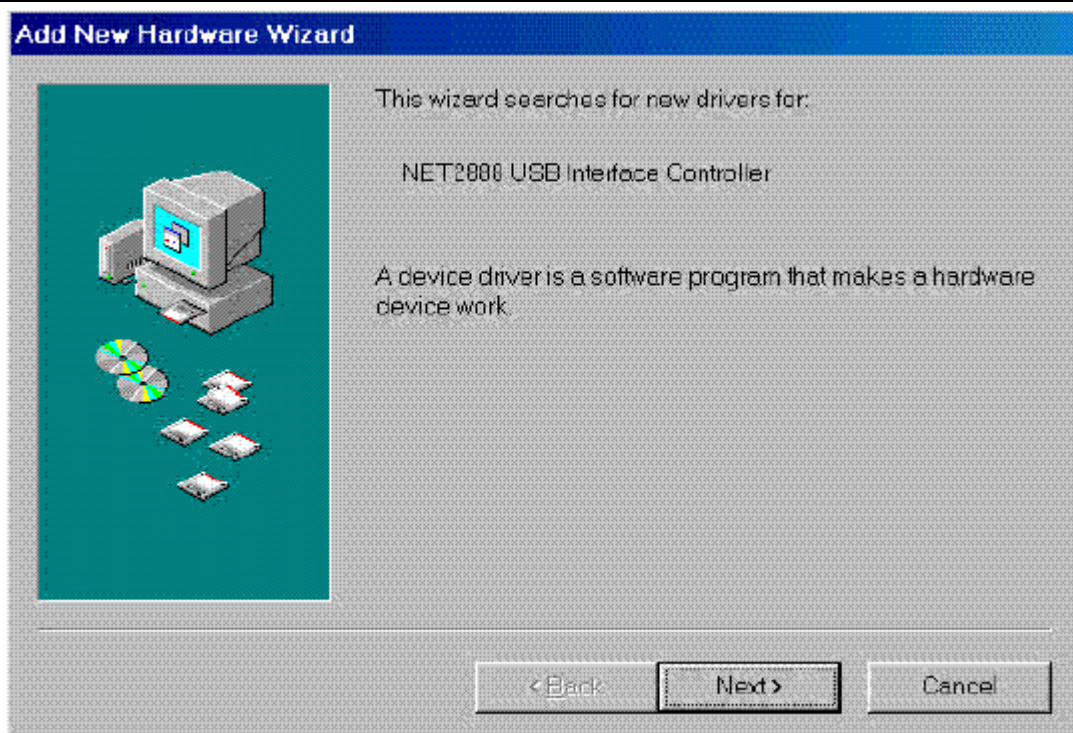
c. 使用非 GALIL 通信软件

DMC-2x00 主串口配置为 DATASET 方式，您的计算机或终端也必须配置为 DATATERM、全双工，无校验，8 个数据位，1 个起始位和 1 个停止位。确认波特率开关设置为上述所需之波特率。

需要将您的计算机配置为“dumb”终端，发送 ASCII 码给 DMC-2x00。

B. 由 USB 通信

注意：GALIL 软件只支持 Windows98，ME 和 2000 下的 USB 口通信。将 USB 电缆连接到计算机和控制器上的 USB IN 口之间，由于在前一步操作中已对控制器上电，因此，控制器将第一连接认作 GALIL USB 控制器。计算机会将 USB 口识别为 NET2888 装置，然后计算机显示屏就显示如下内容：



选择 NEXT，然后选择“Search for the Best Driver for your device”，搜索 GALIL CD-ROM，并进入目录 INSTALL\DMC2000。计算机将识别文件 GALIL USB.INF，此驱动器就让计算机由 USB 与 DMC-2000 控制器进行通信。

一旦安装了 USB 驱动器，由 USB 连接的新 DMC-2000 就会自动进入 GALIL 注册表。每个控制器由其独特的序号来加以区别。如果删除 GALIL 注册表入口，当控制器重新与计算机相连时，就会重新安装。

注意：如果用两个 USB OUT 口，其中之一将 DMC-2000 连接到附加的 USB 装置，计算机就需要 USBHUB.SYS 文件，此文件从 Windows 安装盘安装。

C. 由 Ethernet 通信

a) 用 Windows 版 GALIL 软件

必须在 Windows 注册表中注册控制器，以便主计算机与其通信。通过 GALIL 软件（如 DTERM, DMCTERM, SMARTTERM 或 WSDK）可以进行注册。用 DTERM, DMCTERM, SMARTTERM 在 REGISTRY 菜单下进行注册；用 WSDK，在 FILE 菜单下进行注册。使用 ADD 按钮在注册表中添加一个新入口。选择 DMC-2200 作为控制器类型，输入从您的系统管理器中得到的 IP 地址，选择您想与控制器通信的 UDP 或 TCP 协议相对应的按钮。如果没有分配 IP 地址给控制器，就点击 ASSIGN IP ADDRESS。

ASSIGN IP ADDRESS 将检查连接到网络中的控制器，查看哪一个控制器还没有 IP 地址。然后程序就会提问您是否想将您输入的 IP 地址分配给带有指定序号的控制器。点击 YES，就分配此 IP 地址，点击 NO，移动到下一个控制器，或点击 CANCEL，不保存变更。如果网络上不存在还没有分配 IP 地址的控制器，程序就会给出说明。

做完注册后，点击 OK。如果您不想保存此次修改，就点击 CANCEL。一旦控制器已注册，就从列表中选择正确的控制器，并点击 OK。如果软件成功地与控制器建立了通信，就会在显示屏顶部显示注册表入口。

注意：必须通过 Ethernet 连接来注册控制器。

D. 向终端发送测试命令：

在您连接终端之后，按键盘上的〈Return〉（返回键）或〈Enter〉（回车键）作为应答，控制器以冒号（:）回应。

现在键入

TPA 〈Return〉

此命令让控制器返回 A 轴的当前位置。控制器应以数字回应，例如：

00000000

第 7 步：决定用于正弦波换向的轴：

略

第 8 步：连接放大器和编码器：

当您建立了 GALIL 软件与 DMC-2x00 之间的通信，就可做好准备连接运动控制系统中的其余部件。一般来说，运动控制系统由 PICM-3900/2900 接口模块、各轴放大器及将放大器输出电流转换成运动转矩的电机组成。

PICM-3900 与 DMC 控制器之间由 100pin 高密度屏蔽电缆（Cable-100-1/2/4M）连接。PICM-3900 与放大器的连接方式为每轴一个 15pin D 型连接器，连接极其方便。而 PICM-2900 则采用螺栓端子方式。各引脚定义在后面进行说明。

4 轴以上的运动控制器需要 2 个 PICM-3900/2900 及 2 根 Cable-100-1/2/4M。系统连接过程取决于系统元件和电机类型。

下面是连接一套运动控制系统的前几步：

- A、按放大器说明书正确连接电机到放大器，但不要与 DMC 控制器相连（即不要插入互联电缆）。开通放大器电源，如果放大器工作正常，即使在放大器上电时，电机仍应静止不动。
- B、连接放大器使能信号

由于 PICM-3900 已对 AMPEN 信号进行光电隔离处理，因此，可以很方便地与放大器侧的使能信号相连接。PICM-3900 既可适用于+24VDC 供电的放大器侧使能信号，又可适用于+5V 供电的放大器侧使能信号。确认当伺服侧使能信号是+24VDC 供电时，与该轴相对应的短路棒（JP1~JP4 对应于轴 A~D），务必插入 24G 一侧，而当伺服侧使能信号是+5VDC 供电时，与该轴相对应的短路棒务必插入 5G 一侧。详见后续章节 PICM-3900 说明。

AMPEN 信号受看门狗定时器、电机关断命令 MO 及超差报警使能关断命令 OE1 控制。

控制器侧 AMPEN 信号的标准配置为高电平有效，也就是说，控制器期望放大器使能时，AMPEN 信号为高。当需要将 AMPEN 从高有效改变成低有效时，您可以将 PICM-3900 上的 IC 元件 U2 由 7407 变为 7406 即可。请注意，有一些放大器将此信号称为 INHIBIT。

- C、连接编码器

对于步进电机工作方式，编码器作为选择项，可以连接编码器以读取实时位置值，作为显示之用，但不进行自动闭环补偿。对于伺服电机工作方式，如果您事先规定了正向和负向，请务必确保编码器连线与定义的方向相一致。

编码器是将与电机或机械运动转换成与之相对应的电脉冲信号反馈到控制器。DMC-2x00 接受旋转编码器或光栅反馈信号。典型的编码器提供两路正交脉冲信号，称之为 CHA、CHB，这种编码器称为正交编码器。正交编码器可以是单端（CHA 和 CHB）型或差分型（CHA，CHA-和 CHB，CHB-）。编码器也可以有第三路信号（即定标或一转脉冲），通常用于同步或原点返回操作。

DMC-2x00 可接收带定标脉冲或不带定标脉冲的单端或差分编码器反馈信号。对于编码器刻线密度没有限制，不过反馈到控制器的输入脉冲频率不要超过 3,000,000Hz（即

12,000,000Hz 计数单位), DMC 控制器在内部对接收到的正交脉冲信号进行四倍频处理。例如,如果编码器刻线密度为 10000 周期/inch,最大速度即是 300inch/sec。如果需要更高的编码器频率,请与中宝伦公司联系。

可接收的编码器输出信号标准电平是 TTL (即 0~5V),不过,也可接受电平为 12V 的脉冲信号。(如果使用差分信号,可以将 12V 的脉冲信号直接输入到 DMC-2x00,单端 12V 脉冲信号需要一输入到互补输入端的偏置电压)。

务必确认编码器与 DMC 控制器互联模块之间的连线正确。否则会导致位置反馈计数值不准确或原点返回精度不够。

DMC-2x00 能够接收各轴的模拟反馈信号和脉冲+方向编码器。当使用脉冲+方向编码器时,将脉冲信号连接到 CHA,方向信号连接到 CHB。此时,用户必须用 CE 命令将控制器编码器反馈形式配置成脉冲+方向方式,具体详见本书第二篇命令手册。另外,也可以用其它类型的位置传感器与 DMC 控制器接口,如旋变或绝对值编码器。GALIL 能够为用户定制专用控制器和与之相对应的命令组。如有这方面的特殊需求,请与 GALIL 在中国的代理商——中宝伦公司联系。

D、验证编码器工作正常与否

先从 A 轴编码器开始,在确认连接无误之后,转动电机轴,并用命令 TPA (回车) 查询位置,控制器读回的值就会随电机转动而变化。如果 TPA 读取值不随编码器转动而变化,其原因有如下三种:

- 编码器连线错误——检查相对应轴的连线。
- 编码器损坏——用示波器观察编码器信号。验证 A、B 相幅值在 5~12V 之间。请注意,如果只有一相编码器输出不正常,读取的位置只随一个方向的计数而变化。若编码器损坏,请更换编码器。如果您未能观察到编码器信号,请用不同的编码器试试。
- 控制器硬件损坏——连接同一个编码器到其它轴,如果问题消失,即可判断为控制器硬件故障。请与中宝伦公司联系。

第 9 步 连接伺服电机和步进电机:

A、连接伺服电机

可以将电机和放大器配置成转矩或速度方式。在转矩方式下,放大器增益应该是 10V 指令电压信号对应所需要的最大电流。而在速度方式下,10V 指令电压信号应该对应电机的最大速度。

a. 检查反馈环的极性

假如电机和放大器连接正确,且编码器工作正常,则继续下一步 b。将电机、放大器与控制器连接之前,按下步设定误差限制和转矩限制参数。**注意:**以 A 轴为例。

b. 设定误差限制值以确保安全

通常,关于反馈的正确极性具有不确定性。反馈极性错误会使电机暴走(飞车)。用终端软件,如 DMCTERM,设定下列参数以防系统毁坏:

输入命令:

ER2000 (CR) 设定 A 轴误差限制值为 2000 计数单位

OE1 (CR) 当超差时,断开 A 轴放大器使能

此时,如果电机飞车,位置环误差超过 2000 计数单位,就会立即关断使能信号。

注意:此功能要求控制器 AMPEN 信号与放大器使能信号相连接时才有效。

c. 设定转矩限制值以确保安全

为了限制输出到放大器的最大电压信号,DMC-2x00 控制器有转矩限制命令 TL。此命令设定控制器的最大电压输出,且在初始设置伺服系统时能够用来避免转矩或速度过大。

当放大器以转矩方式工作时,控制器输出电压将直接与电机的转矩输出相关;用户

根据所使用的电机、放大器数据决定这种关系。设定转矩限制值就会限制电机输出转矩。

当放大器以速度方式工作时，控制器的指令输出电压将直接与电机的速度相关。用户根据电机、放大器数据决定这种关系。此时，设定转矩限制值就会限制电机的速度。

例如：以下命令就会将 X 轴指令输出电压限制为 1V：

TL 1 〈CR〉

注意：一旦确定了反馈回路的正确极性，一般来说，就应该将转矩限制值增大到缺省值 9.99。如果转矩限制值在正常工作范围以下，伺服系统就不会正常工作。详见第二篇中的 TL 命令。

d. 连接电机

设定好参数后，即可连接模拟指令信号（ACMD）到放大器。

要想检测反馈极性，用以下命令进行一个运动：

PR1000 〈CR〉 相对位置 1000 计数单位

BGA 〈CR〉 开始 A 轴运动

当反馈极性不正确时，电机就会暴走，但当位置误差超过 2000 计数单位时，控制器就会将使能关断，使电机停止。如果电机暴走，就必须转换回路中的极性。

转换回路极性：

当反馈极性不正确时（正反馈时），用户必须转换回路极性，转换回路极性的方法有多种方法。如果您连接的是 DC 伺服电机，最简单的方法是将两根电枢线进行交换（一般为红色线和黑色线）。如果您连接的是 AC 伺服电机，交换编码器反馈连线即可进行极性转换。如果您用的是单端型编码器，就将 CHA 和 CHB 相交换；如果您用的是差分型编码器，就只交换 CHA+和 CHA-。通过软件命令 MT 和 CE 也能够改变回路极性和编码器极性，详见第二篇。

有时，虽然反馈极性正确（电机不暴走），但运动的方向与期望的方向相反，在这种情况下，需要将电机指令和编码器极性同时转换。

如果电机以所需方向运动，但未能到达目标位置而停止，最可能的原因是由于转矩输出不够，此时，需要检查电机命令输出信号 ACMD，减小电机侧的系统摩擦可能会好一些。用如下命令确认：

TTA 〈Return〉 报告 A 轴转矩

报告实际转矩输出信号的大小，以确认磨擦力的大小。

一旦您建立了闭环系统的正确极性，您就能够调整 PID 滤波器参数 KP, KD, KI。

为了获得良好的系统响应性和高精度，就需要准确调整伺服系统参数。将在下一步描述。

有关控制器与放大器与伺服电机的连接请参见附录中关于 PICM-3900-OPTO 的说明。

B. 连接正弦波换向电机

略

C. 连接步进电机

在步进电机工作方式中，指令脉冲输出信号占空比为 50%。步进电机以开环方式工作，不需要编码器反馈。使用步进电机时，相对应轴的辅助编码器不能用于外部连接。如果使用编码器作为位置反馈，将编码器连接到对应轴的主编码器输入口。所命令的步进位置用命令 RP 或 TD 来查询。编码器位置反馈可以用命令 TP 查询。

能够用滤波器参数 KS 对步进电机脉冲频率进行平滑处理。KS 参数范围为 0.5~8，其中 8 意味着平滑量最大。详见第二篇相关 KS 说明。

所有运动命令均适用于步进电机工作方式，如：PR, PA, VP, CR, JG 等。在参数方面，相对简单一些，使用加减、减速、平滑参数即可。由于步进电机以开环运行，PID 滤波器不起作用，也无位置误差之说。

要将 DMC-2x00 与步进电机相连接，您必须遵循以下步骤进行：

- a. 插入 SM 短路棒
在 DMC-2x00 主板上与有与各轴以步时电机工作方式相对应的短路棒。
- b. 连接脉冲和方向信号到放大器
若使用 PICM-3900-OPTO，直接插入各轴连接电缆即可。其中连接器 J5~J8 对应轴 A~D。
- c. 用 MT 命令将 DMC 控制器配置为步进电机方式，输出脉冲型式分为高有效、低有效脉冲两种。使用 MT2 为低有效脉冲方式，MT-2 为高有效脉冲。详见第二篇中关于 MT 命令的说明。

第 10 步 调试伺服系统：

调整使用伺服电机时所需要的相关参数，使系统补偿得到优化，从而获得快速、精确的响应性能。

滤波器有 3 个重要参数：阻尼系数（微分常数）KD、比例增益 KP、积分常数 KI。以次选择这些参数。

- a. 先将 KI 设为 0

KI 0 〈Return〉 积分常数

并将比例增益和微分常数设定一个较小值，例如：

KP1 〈Return〉 比例增益

KD100 〈Return〉 微分常数

为了增加阻尼就增大 KD（最大值为 4095）。渐渐增大，当电机出现振动（出现异常声音）时，停止增大。此时，若用命令 TEA 〈Return〉 反复发送几次，得到不同的返回值，在出现振荡时，此值的极性交替变化。此时，适当减小 KD，直到系统稳定。

- b. 渐渐增大 KP 值（最大值为 1023），用 TE 命令监测响应性能的改善。例如，

KP10 〈Return〉 比例增益

TEA 〈Return〉 监测返回值

随着 KP 增大，误差会减小。

如果增益过大，也会出现系统振荡，此时，适当减小 KP。一般来说，KP 不应该大于 KD/4。

- c. 选择 KI，从 0 值开始渐渐增大。积分器会消除位置误差，使系统控制精度得以提高。用 TEA 〈Return〉 进行验证，最终结果是随着 KI 增大，误差变为 0。如果 KI 过大，会导致系统振荡。在系统出现振荡时，适当减小 KI。

其余各轴依次按上述方法进行调整。

用户也可以用可选择的 WSDK 伺服参数自动调整软件对各轴进行参数调整。

3. 设计编程入门

在对系统进行连接、基本参数配置之后，我们举一些例子作为设计编程的入门，以便大家能够对简单运动编程有个初步了解。

1) 系统配置

指令	解释
KP10, 10, 10, 10	对 A~D 轴设置增益 KP
KP=10	对所有轴设置增益 KP 的另一种方法
KPA=10	对 A 轴设置增益
KP, 20	只对 B 轴设置增益

指令	解释
----	----

OE1, 1, 1, 1, 1, 1, 1, 1	所有轴设置为超差时使能自动断开
ER=1000	所有轴误差限制值设为 1000 计数单位
KP10, 10, 10, 10, 10, 10, 10, 10	对 A~H 轴设置增益
KP=10	对 A~H 轴设置增益的另一种方法
KP,, 10	只对 C 轴设置增益
KPD=10	对 D 轴设置增益的另一种方法
KPH=10	对 H 轴设置增益的另一种方法

2) 包络线运动

使 A 轴以速度为 20, 000 计数单位/sec, 加速度、减速度为 100, 000 计数单位/sec² 运动 10, 000 计数单位的距离, 此例中, 电机起动、运转、停止

指令	解释
PR10000	距离
SP20000	速度
AC100000	加速度
DC100000	减速度
BGA	开始运动

3) 多轴运动

4 个轴分别运动

指令	解释
PR500, 1000, 6000, -400	A、B、C、D 轴的距离
SP10000, 12000, 20000, 10000	A、B、C、D 轴的速度
AC10000, 10000, 10000, 10000	A、B、C、D 轴的加速度
DC80000, 40000, 30000, 50000	A、B、C、D 轴的减速度
BGAC	开始 A、C 轴运动
BGBD	开始 B、D 轴运动

4) 独立运动

分别指定运动参数, 如下:

指令	解释
PR, 300, -600	B、C 轴距离
SP, 2000	B 轴速度
DC, 80000	B 轴减速度
AC, 100000	B 轴加速度
AC,, 100000	C 轴加速度
DC,, 150000	C 轴减速度
BGC	开始 C 轴运动
BGB	开始 B 轴运动

5) 位置查询

用 TP 命令对 4 个轴的实际位置进行查询

指令	说明
TP	查询所有 4 个轴的实际位置
TPA	查询 A 轴的实际位置
TPB	查询 B 轴的实际位置
TPC	查询 C 轴的实际位置
TPD	查询 D 轴的实际位置

用 TE 命令查询位置误差（指令位置——实际位置）

指令	说明
TE	查询所有轴的位置误差
TEA	查询 A 轴的位置误差
TEB	查询 B 轴的位置误差
TEC	查询 C 轴的位置误差
TED	查询 D 轴的位置误差

6) 绝对位置

目的：熟悉指定绝对位置的命令

指令	说明
DP0, 2000	定义 A、B 轴当前位置为 0, 2000
PA7000, 4000	设置想要的绝对位置
BGA	开始 A 轴运动
BGB	开始 B 轴运动

在两个运动完成后，命令 A、B 轴回到零点：

PA0, 0	移动到 0, 0
BGAB	开始 A、B 轴运动

7) 速度控制

目的：驱动 A、B 轴电机以指定速度运动：

指令	说明
JG10000, -20000	设定 A、B 轴的 JOG 速度和方向
AC100000, 40000	设定 A、B 轴的加速度
DC50000, 50000	设定 A、B 轴的减速度
BGAB	开始 A、B 轴运动

在数秒之后，命令：

JG-40000	设定 A 轴新速度和方向
TVA	读取 A 轴速度

然后：

JG, 20000	设定 B 轴新速度
TVB	读取 B 轴速度

以上试验会使速度和方向发生变化。用命令 ST 能够停止运动

ST	停止运动
----	------

8) 在转矩限制条件下工作

电机命令的输出幅值可以用命令 TL 进行限制。

指令	说明
TL0.2	设置 A 轴输出限制为 0.2V
JG10000	设置 A 轴速度
BGA	开始 A 轴运动

在本例中，在放大器以转矩方式工作时，由于输出信号不足以克服摩擦力，A 轴电机有可能不会运动。如果开始运动，用手指就能轻松地使电机停止。

用命令 TL 渐渐增大转矩命令输出，如：

指令	说明
TL1.0	增大转矩限制值到 1V



在输入完上述命令之后，退出编辑器方式，用〈Ctrl〉Q。要想开始执行程序，用命令

XQ#A	执行程序#A
------	--------

执行程序#B

重复执行#B 程序

C

标号#C

EN

结束

要想执行此程序，命令：

XQ # A

执行程序#A

此程序使 A 轴移动到初始位置 4000，并将其读取回，同时将其值的 1/2 作为新的目标位置。

注意：-TPA 是读取 A 轴位置值的内部变量。其使用方法是在 DMC 命令前加下划线。

15) 直线插补

目的：分别使 A、B、C 轴以直线插补方式移动 7000，3000，6000 的距离。一般来讲，电机一起启动、停止。

指令

说 明

LMABC

指定 A、B、C 轴为直线插补运动

LI7000, 3000, 6000

直线插补相对距离

LE

直线插补结束

VS6000

矢量速度

VA20000

矢量加速度

VD20000

矢量减速度

BGS

开始运动

16) 圆弧插补

目的：以圆弧插补方式移动 A、B 轴形成下图所示的运动轨迹。注意：矢量运动从位置 (0, 0) 点开始，定义在任何矢量运动程序的开始处。详见后续“应用编程”一章。

指令

说 明

VMAB

选择 A、B 轴为圆弧插补运动

VP-4000, 0

直线线段运动

CR2000, 270, -180

圆弧线段运动

VP0, 4000

直线线段运动

CR2000, 90, -180

圆弧线段运动

VS1000

矢量速度

VA50000

矢量加速度

VD50000

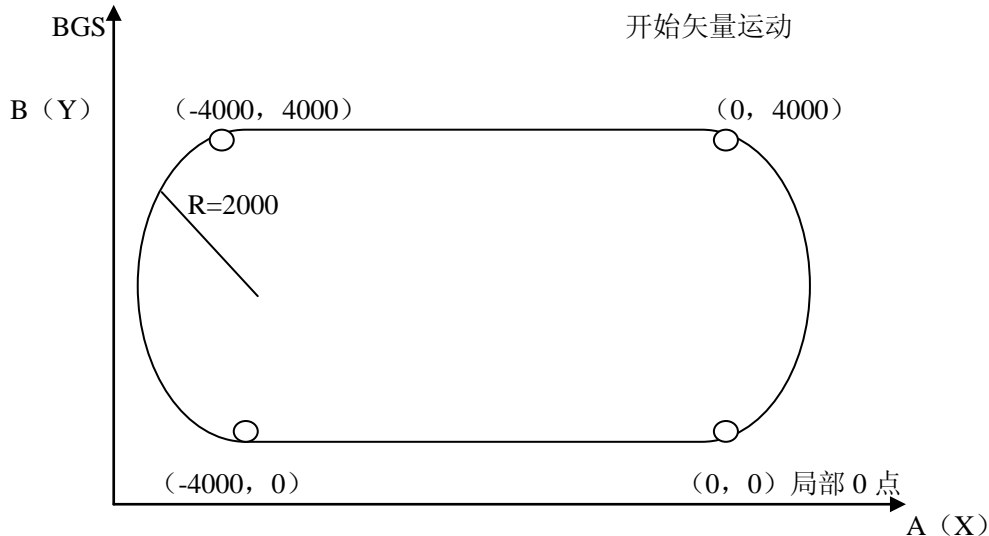
矢量减速度

VE

矢量程序结束

BGS

开始矢量运动



第三章 硬件接口说明与连接

1. 概述

中宝伦公司为了适应中国用户对低价格产品的迫切需求，结合中国用户实际使用情况，对美国 GALIL 公司外围配套件进行本地化设计、制造。最新推出的升级版互联模块 PICM-3900-OPTO 可以适用于任何一款具有 100pin 输出连接器的 DMC 控制器。对与控制器相连接的全部正向限位、负向限位、原点开关、急停、报警及通用输入/输出信号进行光电隔离处理。详见有关 PICM-3900-OPTO 的说明。

对于 GALIL DMC 高档型控制器，均提供各轴正/负限位、原点返回、急停、报警、使能、通用输入/输出（8 入/8 出（1~4 轴）、16 入/16 出（5~8 轴））以及可接受±10V 电压的 8 路 A/D 输入接口。

本章主要就输入、输出功能进行说明，具体连接方法请参见有关 PICM-3900-OPTO 的说明。对于可扩展的 64 点 I/O，请参见有关 I/O 扩展模块 IOM-1964 的说明。

若您需要使用辅助编码器功能，也有相应的互联模块 ICM-2908 及连接电缆 CABLE-36 供选用。

2. 输入接口

2.1 限位开关输入

正向限位开关（FLSX）一旦被压下而生效，就会立即禁止机械体向正方向运动；而负向限位开关（RLSX）一旦被压下而生效就会立即禁止机械体向反方向运动。如果在运动期间，限位开关生效，控制器就会以 DC 命令所设置的减速度使运动减速而停止。在限位开关生效后，不断开使能，电机虽然停止，但仍处于闭环状态，使电机位置保持在锁定状态。当正、反向限位开关生效时，当前正在运行的程序被中断，控制器会自动跳转到#LIMSWI 子程序。用户可以将此子程序包含在任何一个运动控制子程序中，一旦限位开关生效，对于执行所规定的命令非常有用。详见第 6 章关于自动子程序的说明。

在限位开关生效后，如果限位开关的逻辑状态未返回到无效状态，就不可能使电机向限位开关所保护的方向进一步运动。此开关涉及常开、常闭两种状态。在逻辑状态回复之前的任何运动均会导致以下报警：“022-Begin not possible due to limit switch”。

内部变量-LFX 和-LRX 分别含有正、反向限位开关的状态。（X 代表轴 A、B、C、D 等）。内部变量值是 0 或者 1，对应于限位开关的逻辑状态。使用终端程序，用命令 MG-LFX 或 MG-LRX 可以将限位开关的状态显示在屏幕上。也能够用 TS 命令查询限位开关的状态。详见本书第二篇中有关 TS 命令说明。

2.2 原点开关输入

原点开关输入用于机械原点返回操作，原点开关输入状态的变化提示控制运动体到达了一个特殊参考点。参考点可以是空间或编码器定标脉冲中的一点。

原点输入检测此开关状态的变化，每次变化非 0 即 1，属逻辑电平。此变化会使控制器执行用户所指定的原点返回程序。

GALIL 控制器支持 3 种原点返回方式：寻找原点开关（FE）、寻找标志脉冲（FI）和标准原点返回方式（HM）。

寻找原点开关程序用如下命令来进行：

FEA <Return>

BGA <Return>

此程序会使电机加速到恒定速度运动，一旦检测到原点开关输入逻辑状态发生变化，就使电

机减速停止，此点即为原点。FE 运动的方向取决于原点开关的状态设定，用户事先可以用命令 AC、DC、SP 设置加速度、减速度和速度。**建议使用大的减速度，以便在检测到原点开关输入信号后，电机能够快速停止。**

寻找标志脉冲程序用如下命令：

FIA 〈Return〉

BGA 〈Return〉

此程序会使电机加速到用户所指定的速度运动，一旦控制器检测到标志脉冲信号由低变高，电机就减速停止。电机的回零速度和方向用 JG 命令来指定，加速度、减速度分别用 AC、DC 命令来设置。尽管寻找标志脉冲是原点返回操作的选择，与原点开关输入的逻辑状态变化无关，但与标志脉冲信号的电平变化密切相关。

标志原点返回方式用如下命令来进行：

HMA 〈Return〉

BGA 〈Return〉

此方法是以上两种方法的结合。电机以所设定的速度运动，一旦检测到原点开关输入逻辑状态变化，电机就会减速停止，然后，电机就会改变方向，以 256 计数单位/sec 的速度逼近原点开关，当此开关的逻辑状态再次变化时，电机以同样的速度向正向运动（计数值增大的方向），控制器一检测到标志脉冲（Index），就命令电机减速停止，并将此位置定义为 O 点（机械原点）。可以用命令 MG-HMA 查询原点开关输入的逻辑状态。如果逻辑状态是低或者高，其返回值分别为 0 或 1。原点开关输入状态也可以用命令 TS 来查询。

相关原点返回操作的进一步说明及举例请参考第二篇中的 HM、FI、FE 命令及本篇中的“编辑运动程序”一节。

2. 3 急停输入

急停输入的功能是一旦此输入点逻辑状态发生变化就立即使控制器停止运行。

注 1：急停输入的响应速度远远不同于限位开关输入有效时的情况。当急停输入生效时，控制器立即停止执行运动命令，而限位开关输入生效时，控制器会使电机减速后停止。

注 2：急停输入的作用与各轴超差关断使能的状态设置有关。如果每个轴超差关断使能有效，当急停信号发生时，就会关断该轴使能。此时，由于伺服系统不再处于闭环控制之下，所以电机依靠惯性停止。如果超差关断使能设置为无效，电机就会尽可能快地减速停止，此时，电机仍然保持在伺服闭环状态。

当检测到急停输入状态变化时，控制器就终止当前正在运行的所有运动程序。进一步的说明请参见第二篇中的 OE 命令。

2. 4 复位输入

此输入为低电平时，控制器将执行复位操作。

2. 5 通用输入

DMC-2x00 有 8 个通用输入点，能够用函数@IN[X]分别读取这些输入点状态，其中 X 代表输入点号（1~8）。这些输入点属未定义输入点，用户可以用这此输入点建立与外部事件相关的条件语言。

对于 5~8 轴控制器，有 16 个通用输入点。

2. 6 模拟输入

DMC-2x00 等高档控制器有 8 路模拟输入，输入电压范围为-10V~+10V。分辨率为 12bit ADC，即大约为 0.005V。16bit ADC 作为选项，订货时需要单独订购；输入阻抗为 10K Ω 。模拟输入规定为 AN[x]，其中 x 是通道号（1~8）。

2. 7 TTL 电平输入信号

辅助编码器输入信号为差分型 $\pm 12V$ 输入信号，通常配置为接收 TTL 电平输入信号。

对于 DMC-2x00 来说, 这些输入信号也能够当做通用输入信号使用。每轴由两路输入 CHA 和 CHB 构成。辅助编码器输入分配为输入点 81~96。用函数@IN[81]~@IN[96]存取这些输入点的状态。

注意: 若将某轴配置为步进电机工作方式, 这些输入就不能当做辅助编码器输入使用。

2. 8 输出信号

DMC-2x00 提供专用和通用输出功能

2. 8. 1 通用输出

DMC-2x00 提供 8 点通用输出信号, 这些输出信号虽然为 TTL 电平, 但当使用 PICM-3900-OPTO 时, 均对这些信号采取光电隔离处理, 这 8 个点定义为 OUT1~OUT8。用命令 SB (设置位)、CB (清除位)、OB (输出位)、OP (输出位) 将这些输出点开通或关断。可以用内部变量-OP 和函数@OUT[x]来检查输出点的状态。

注意: 由于 PICM-3900-OPTO 提供光电隔离输出, 此时, 需要用户提供隔离型电源 (+5V~+24VDC)。具体连接详见附录 PICM-3900-OPTO 说明。

2. 8. 2 报警输出

当控制器出现报警时, 红色 LED 灯点亮, 报警输出 ERROR 变低。以下几种原因可能会导致控制报警:

- a. 位置超差;
- b. 控制器侧复位变低或受干扰影响;
- c. 控制器损坏, 处理器处于自身复位状态;
- d. 驱动报警输出的 IC 芯片损坏

2. 8. 3 比较输出 CMP

输出比较由控制器主编码器的位置反馈值与 OC 命令所设置的比较值相比较而产生。每当反馈值跨越比较值时, 在 CMP 端产生一个低脉冲 (1 μ sec)。进一步信息请参考《〈命令手册〉》中 OE 的说明。

2. 8. 4 使能输出

当使用 PICM-3900-OPTO 时, AMPEN 被光电隔离处理, 输出形式为 OC 门, 因此, 可以很方便地与各种放大器进行接口 (TTL 电平或触点型 (24V 开关信号))。详见附录相关 PICM-3900-OPTO 的说明。

2. 8. 5 模拟指令输出 ACMD

其输出信号为 0~ ± 10 V 模拟电压指令, 分辨率为 16bit DAC。

3. 扩展 I/O

DMC-2x00 提供 64 点 I/O 扩展功能, I/O 扩展模块型号为 IOM-1964。进一步说明参见 IOM-1964 说明。

通过将 XON 开关设置为 ON 就能够使软件握手有效, 在此方式中, 控制器产生将产生/接受 XON 和 XOFF 字符来控制来自/到终端的字符流。控制器对 XOFF 字符使用十六进制值\$13, 而对 XON 字符使用十六进制值\$11。

DMC-2x00 的辅 RS232 口可以被配置为通用串口或菊花链 (只对 DMC-2000)。当配置成通用口时, 就能够命令此口发送 ASCII 信息到另一台 DMC-2x00 控制器或者到显示终端或面板。

配置串口 2 的命令格式为:

CCm, n, r, p

其中 m 设置波特率, n 设置握手或无握手方式, r 设置通用口或辅口, p 使回声开或关。

m——波特率——300, 1200, 4800, 9600, 19200, 38400

n——握手: 0=No, 1=Yes

r——方式：0=通用口，1=菊花链

p——Echo：0=OFF，1=ON；若 r=0 时才有效。

注：对于辅口握手来说，RTS 和 CTS 的作用相反。

举例：

CC1200, 0, 0, 1 配置辅通信口为 1200 波特，无握手，通用口方式，Echo 为 ON。

菊花链（只对 DMC-2000）

在菊花链中，可以连接 8 台 DMC-2000 控制器。此时，允许从一个串口对多台控制器发送命令。一台 DMC-2000 经由 RS232 口 1 或主口连接到主终端。然后将该台控制器的口 2 或辅口与下一台 DMC-2000 的口 1 相连，以此类推。通过设定控制器前面的 3 个地址拨码开关（A0, A1, A2）来配置每台 DMC-2000 的地址。

当多台控制器连接到菊花链中时，控制器之间的电缆两头的连接器应该均为孔型。

ADR1 代表 2^0 位，ADR2 代表 2^1 位，ADR4 代表 2^2 地址位。8 个可用地址 0~7 如下：

A2	A1	A0	地址
OFF	OFF	OFF	0
OFF	OFF	ON	1
OFF	ON	OFF	2
OFF	ON	ON	3
ON	OFF	OFF	4
ON	OFF	ON	5
ON	ON	OFF	6
ON	ON	ON	7

要与其中任何一台 DMC-2000 通信，用命令“%A”，其中 A 是该台控制器的地址。跟随其后的所有命令都只发送到带有该地址的控制器。只有当给出一个新“%A”命令时，才会发送命令给出另一台控制器。唯一例外是“!”命令。要同时与菊花链中的所有 DMC-2000 控制器进行通信，在软件命令之前插入字符“!”，所有控制器均接收命令，但只有地址 0 会回应。

注意：要想配置每个控制器的口 2。必须指定 CC 命令。

举例 1：菊花链

目的：用两台控制器构成 7 轴运动系统，其中一台为 DMC-2040，另一台为 DMC-2030，地址 0 是 DMC-2040，地址 1 是 DMC-2030。

地址 0（DMC-2040）

A 轴运动 500 计数单位

B 轴运动 1000 计数单位

C 轴运动 2000 计数单位

D 轴运动 1500 计数单位

地址 1（DMC-2030）

A 轴运动 700 计数单位

B 轴运动 1500 计数单位

C 轴运动 2500 计数单位

4. 控制器对数据的响应

DMC-2×00 控制器对于有效命令返回冒号 (:), 而对无效命令返回问号 (?),

例如，如果对 BG 命令以小写发送，DMC-2×00 就会返回？

bg <return>

无效命令，小写

?

DMC-2×00 返回？

当控制器收到无效命令时，用户可以查询错误代码，错误代码会指出出现无效命令响应的原因。要查询错误代码，键入命令 TC1，例如：

```
? TC1    <return>    报告代码命令
1 Unrecognijed        返回的响应
```



接收无效命令响应有多种原因，最常出现的原因是：不认识的命令（印刷条目或小写）；给出时间不合适（例如在运动期间）；或超出命令值范围（例如超出最大速度）。所有出错代码列表见第二篇中 TC 命令章节。

5. 查询控制器

5. 1 查询命令

DMC-2×00 有一组直接查询控制器的命令，当输入命令在回车和行进下行处，以十进制格式返回所要查询的数据。所返回数据的格式能够用位置格式(PF)、变量格式(VF)和前充零(LZ)命令进行修改。详见第 7 章和本书第二篇。

5. 2 查询命令汇总

命 令	说 明	
RP	报告命令位置	
RL	报告锁存位置	
^R^V	控制软件版本信息	
SC	停止代码	
TB	告诉状态	
TC	告知错误代码	
TD	告知第二编码器反馈值	
 TE	读取位置误差	
TI	读取输入状态	
TP	读取位置值	
TR	读取轨迹	
TS	读取开关状态	
TT	读取转矩值	
TV	读取速度值	

例如，下面的例子说明如何显示 X 轴当前位置值：

```
TPA <return>    读取 A 轴位置
0000000000      控制器响应
TP AB < return >    读取 A B 轴位置
0000000000,0000000000    控制器响应
```

5. 3 查询当前命令值

大多数命令均能用问号（？）作为轴指定器进行查询。对想要查询的各轴键入用？号跟随的命令，如：

```
PR ?,?,?/?    查问 A, B, C, D 轴的位置值
PR ?          查问 B 轴的位置值
也能够用操作数对控制器查问。
```

5. 4 操作数

大多数 DMC-2×00 命令均有可用于查问相对应的操作数，操作数必须用于有效的 DMC 表达式里。例如，要显示操作数的值，用户可以用命令：MG “Operand” 其中“Operand” 是有效 DMC 操作数。

所有命令操作数均以下划线符 (_) 开始。例如，将 A 轴当前位置值用命令分配给变量“V”

V=_TPA

本书第二篇命令手册中解释了有等效操作数的所有命令。也可参见第 7 章中关于操作数的描述。

5. 5 命令汇总

关于整个命令汇总，请参见本书第二篇。

第四章 通 信

1. 说明

DMC-2000 有两个 RS-232 口、一个 USB 输入口及两个 USB 输出口，而 DMC-2100 和 DMC-2200 分别有两个 RS-232 口和 Ethernet 口。其中，主 RS-232 是数据设置，通过前面板上的开关进行配置。辅 RS-232 口是数据终端，可以用软件命令 CC 进行配置。辅 RS-232 口可以配置成菊花链工作方式（只对 DMC-2000）成通用通信口。此配置可以用 BN 命令加以保存。RS-232 口也有时钟同步线，使多台控制器同步运动。

2. RS-232 口

主、辅 RS-232 口的引脚定义如下所示。**注意：**除输入与输出相反之外，辅 RS-232 口基本上与主 RS-232 口的引脚定义相同。DMC-2×00 也可以在出厂前配置为 RS-422 方式。其引脚定义也列其后。

注意：如果您将辅 RS-232 口连接到是 DATASET 类型的终端设备上，就需有要使用连接器适配器，将 dataset 变为 dataterm 型。此电缆也称为‘NULL’modem 电缆。

A. 主 RS-232 (P1) DATA TERM

脚号	信 号 说 明	脚号	信 号 说 明
1	CTS---输出	6	CTS---输出
2	传送数据---输出	7	RTS---输入
3	接收数据---输入	8	CTS---输出
4	RTS---输入	9	NC （可以连接到+5V 或采样时钟）
5	GND		

B. 辅 RS-232 (P2) DATASET

脚号	信 号 说 明	脚号	信 号 说 明
1	CTS---输入	6	CTS---输入
2	传送数据---输入	7	RTS---输出
3	接收数据---输出	8	CTS---输入
4	RTS---输出	9	+5V （可以用短路棒连接到采样时钟）
5	GND		

C. 主 RS-422 (P1)

脚号	信 号 说 明	脚号	信 号 说 明
1	CTS---输出	6	CTS+---输出
2	发送数据---输出	7	发送+---输出
3	接收数据---输入	8	接收+---输入
4	RTS---输入	9	RTS+---输入
5	GND		

D. 辅 RS-422 (P2)

脚号	信 号 说 明	脚号	信 号 说 明
1	CTS---输入	6	CTS+---输入
2	接收数据---输入	7	接收+---输入
3	发送数据---输出	8	发送+---输出
4	RTS---输出	9	RTS+---输出
5	GND		

E. RS-232 配置

将您的 PC 机配置成 8 个数据位，1 个起始位，1 个停止位，全双工，无校验方式。

可以用前面板上的拨码开关选择 RS-232 通信口的波特率。如下：

开 关 设 定			
9600	19. 2	3800	波 特 率
ON	ON	OFF	1200
ON	OFF	OFF	9600
OFF	ON	OFF	19200
OFF	OFF	ON	38400
OFF	ON	ON	115200

握手方式：

可以将主 RS-232 口配置成硬件或软件握手方式。对硬件握手而言，将 HSHK 开关设为 ON，在此方式中使用 RTS 和 CTS 线。无论何时 DMC-2×00 不准备接收附加字符，CTS 线会变高。RTS 线会禁止 DMC-2×00 发送附加字符。注意，RTS 线为了禁止而变高。为了确保高波特率通信正确，应该将此开关设为 ON。

命 令

%0

PR500,1000,2000,1500

%1

PR700,1500,2500

! BG

说 明

只对控制器 0 通信 (DMC-2040)

指定 A,B,C,D 轴距离

只对控制器 1 通信 (DMC-2030)

指定 A,B,C 轴距离

开始两台控制器运动

菊花链中的同步采样时钟：

可以对菊花链中的所有 DMC-2000 控制器的采样时钟进行同步。与计算机相连的第 1 台控制器应该在 JP3 上插入一个短路棒，使标号为 S 和 8 的引脚相连接。注意：此连接需要一个放在旁边的短路棒。子控制器应该在 JP3, JP4 上分别插入短路棒，使两处标号为 S 和 8 的引脚相连接。

3. Ethernet 配置 (DMC-2100/2200)：

3. 1 通信协议

Ethernet 是以邮袋为单位传送信息的区域网络。因此，需要通信协议来规定如何发送、接收这些邮袋。DMC-2100/2200 支持两种工业标准协议，TCP/IP 和 UDP/IP。控制器以所接触的格式自动响应。

TCP/IP 是一种“应答”协议。为了开始通信，必须将主站与从站相连。当收到发送的邮袋时，会告知已收到。若未收到，告知回应，就认为此邮袋信息丢失，并需要重新发送。

与 TCP/IP 协议不同的是，UDP/IP 不需要“应答”。此协议类似于用 RS-232 进行通信。如果丢失信息，控制器不返回冒号或问号。由于此协议不提供丢失信息，因此，发送方必须重发邮件。

尽管 UDP/IP 更为高效，简单，但 GALIL 建议使用 TCP/IP 协议。TCP/IP 确保如果在传送当中，邮袋被丢失或破坏，就将重发这些邮袋。

Ethernet 通信以邮袋传送信息，邮袋必须限定在 470 数据字节以内。更大的邮袋可能引起控制器通信阻塞。

注：为了不丢失传送信息，GALIL 推荐用户在发送下一个邮袋之前等待邮袋接收的回答信息。

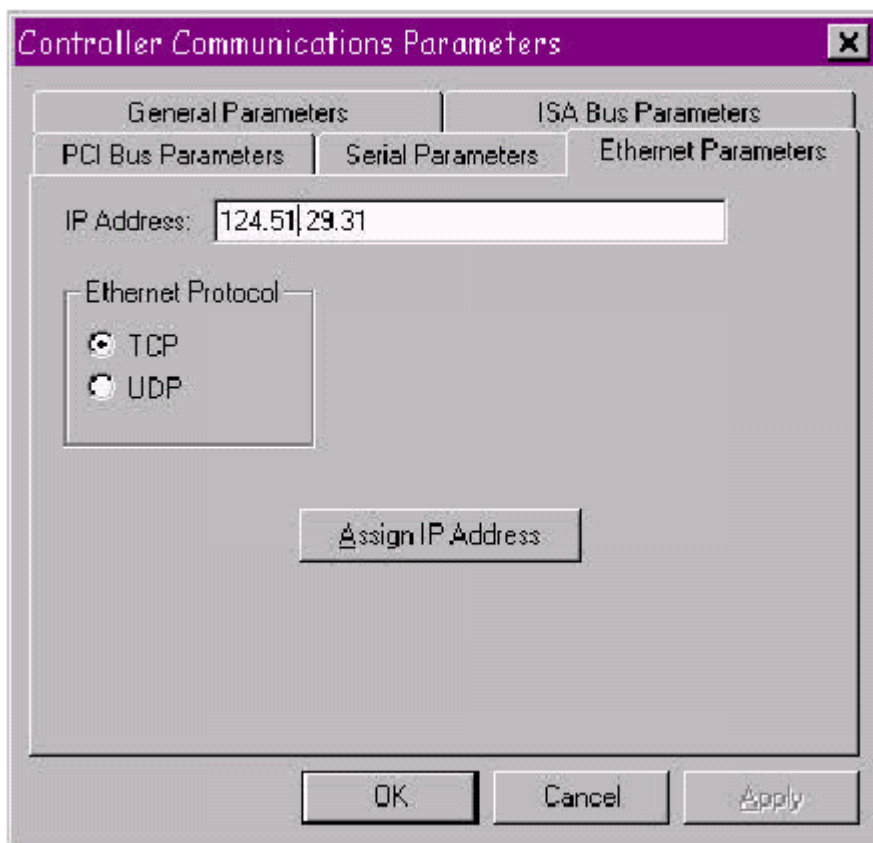
3. 2 地址

有三级地址定义 Ethernet 装置。第一级是 Ethernet 或硬件地址。这是一个唯一且永久的 6 字节数。任何装置之间不会有相同的 Ethernet 地址。DMC-2100/2200 Ethernet 地址由工厂设置，此地址的最后两个字节是控制器的序列号。

第二级地址是 IP 地址。它是一个 32bit（既 4 字节）数。IP 地址由各局域网络规定，且必须局域性地加以分配。用多种方法对控制器分配 IP 地址。

第一种方法是使用由 Ethernet 连接的 BOOT-P 软件工具（DMC-2100/2200 必须连接到网络中并上电）。关于 BOOT-P 的简要说明参见“第三方软件”一节。可以使用内部网络上的 BOOT-P 服务器或 GALIL 终端软件。要使 GALIL BOOT-P 工具软件，就在终端仿真器中选择注册。选择 DMC-2100/2200，然后选择 Ethernet 参数表，在提示行输入 IP 地址，并选择 TCP/IP 或 UDP/IP 作为通信协议。至此，点击 ASSIGN IP ADDRESS。GALIL 终端软件就会以当前设有 IP 地址的网上所有控制器的列表予以回应；用户选择控制器，软件就会给控制器分配所指定的 IP 地址。然后，进入终端软件，键入 BN 并回车以保存 IP 地址到控制器的非易失性存储器中。

注意：确保只有一个 BOOT-P 服务器在运行。如果您的网络有 DHCP 或 BOOT-P 在运行，它就可以自动分配 IP 地址给连接到网络中的控制器。为了确保 IP 地址正确无误，在把控制器连接到 Ethernet 网络之前，请与您的系统管理员联系。



第二种设置 IP 地址的方法是通过 DMC-2100/2200 主 RS-232 口发送 IA 命令，输入要分配的 IP 地址，地址为 4 字节数，中间用逗号分开(工业标准就这样使用)或带符号 32 位数(例如，IA124, 51, 29, 31 或 IA2083724575)，输入 BN 并回车，将 IP 地址保存到控制器的非易失性存储器中。

注意：GALIL 特别推荐，所选择的 IP 地址不是能够通过 GATEWAY 存取的地址。GATEWAY 是管理内部网络和外部世界通信的应用软件。

Ethernet 地址的第三级是 UDP 或 TCP 接口号。GALIL 控制器不需要指定的接口号。每次连接控制器时，接口号由顾客或主站建立。

3.3 与多装置通信

DMC-2100/2200 能够支持多主多从通信，主装置可以是多台 PC 机，发送命令给控制器；从装置典型的外围 I/O 装置，从控制器接收命令。

注意：“ Master”等效于互联网 “Client”，“Slave”等效于互联网 “server”。

Ethernet 端口是在一台装置内的通信源。**DMC-2100/2200 同时最多能够有 6 个 Ethernet 端口开放。**使用 TCP/IP 时，每个主或从使用各自的 Ethernet 端口。在 UDP/IP 中，一个端口可以用于所有主装置，但一个从装置使用一个端口(Pings 和 ARP 不占用端口)。如果所有 6 个端口均在使用中，且第 7 个主装置试图接入，就会发送“reset packet”，在其窗口应用框中产生相应的报警。

注意：有许多方法对控制器进行复位，硬件复位（按下复位按钮或对控制器断电）及软件复位（键入 RS，通过 Ethernet 或 RS232 发送）。不会使控制器断开连接的唯一复位是由 Ethernet 进行软件复位。

当 GALIL 控制器作为主装置使用时，用 IH 命令来分配端口并连接到它的从装置。IP 地址按 4 字节数，中间用逗号分开输入（工业标准就是如此定义），或者输入带符号 32 位数。也可以指定接口号，但如果不指定，其缺省值为 502。要使用的协议（TCP/IP 或 UDP/IP）也必须在此时指定。否则，控制器就不会连接到从装置。（例如：IHB=151, 25, 255, 9<179>2，这样就会打开端口#2，

并连接到 IP 地址 151, 25, 255, 9, 接口号为 179, 使用 TCP/IP 协议)

为了与 I/O 装置对话, 可以用附加协议层。Modbus 是 RS485 协议, 它对信息以二进制邮袋进行打包, 作为部分 TCP/IP 邮袋进行发送。在这个协议中, 各从装置有 1 字节从地址。DMC-2100/2200 对端口能使用指定的从地址或缺省地址。

Modbus 协议有一组称之为功能码的命令。DMC-2100/2200 支持 10 个主要功能码:

序号	功能码	定 义
1	01	读线圈状态 (读位)
2	02	读输入状态 (读位)
3	03	读保持寄存器 (读字)
4	04	读输入寄存器 (读字)
5	05	强行设置单线圈 (写 1 位)
6	06	预置单寄存器 (写 1 个字)
7	07	读其它状态 (读错误码)
8	15	强行设置多线圈 (写多个位)
9	16	预置多个寄存器 (写多个字)
10	17	报告从 ID

DMC-2100/2200 提供三级 Modbus 通信。第一级允许用户创建原邮袋并接收原数据。它使用 MBh 命令, 功能码为-1。命令格式为:

MBh=-1, Len, array[]

其中: Len 是字节数

array[]是含数据的数组

第二级包含 Modbus 结构。这需要发送配置和特殊命令给 I/O 装置。格式变化取决于调用的功能码。更为详细的说明请参考第二篇命令手册。

Modbus 通信的第三级使用标准 GALIL 命令。一旦配置为从装置, 可以使用的命令有: @IN[], @AN[], SB, CB, OB 和 AO。例如, AO2020, 8.2 就会告诉 I/O 号 2020 输出 8.2V 电压。

如果不需要规定的从地址, 要使用的 I/O 号能够用下式求得:

$$\text{I/O 号} = (\text{Handle Num} * 1000) + ((\text{Module} - 1) * 4) + (\text{BitNum} - 1)$$

其中: Handle Num 是端口号。从 1 (A) ~6 (F)。Module 是模块在插槽中的位置, 从 1~16。BitNum 是模块中的 I/O 点, 从 1~4。如果使用明确的从地址, 方程式变为:

$$\text{I/O 号} = (\text{Slave Address} * 1000) + (\text{Handle Num} * 1000) + ((\text{Module} - 1) * 4) + (\text{BitNum} - 1)$$

要想了解与 OPT0-22 机箱进行通信的例子, 请参考附录。

哪个装置接收来自控制器的什么信息取决于许多因素。如果一个装置对控制器发问, 如果它不告诉控制器将此发问发送到另一个装置, 它就会收到响应。如果产生响应的命令是部分下载的程序, 响应就会按路线图传送到由 ENET 开关缺省规定的那个口(如果没有准确告诉去另一个口), ENET 开关“ON”指定 Ethernet 在进入最后端口与控制器进行通信, “OFF”指定主 RS232。要对信息指定规定的目的地, 就将{Eh}添加到命令的尾部。(例如, MG{EC} “Hello”就将信息“Hello”发送到端口#3。TP , , ? {EF}将 Z 轴位置发送到端口#6)。

3. 4 多点发送

多点发送只用于 UDP/IP, 它类似于广播站(处于网络上的每个人都会获得信息)。但指定给一组。换言之, 在所指定组内的所有装置都会接收到以多点方式发送的信息。网络上可以有许许多多点发送组, 由其多点发送 IP 地址加以区分。要想与处于一个指定的多点发送组中的所有装

置进行通信，就可以将信息发送到多点发送 IP 地址而不是各个装置 IP 地址，所有 GALIL 控制器的多点发送缺省地址为 239, 255, 19, 56，用 IA>u 命令能够更改控制器的多点发送 IP 地址。

3. 5 使用第三方软件

GALIL 控制器支持 ARP, BOOT-P 和 Ping 等软件工具来建立 Ethernet 连接。ARP 是一个应用软件。用它来以指定的 IP 地址确定装置的 Ethernet（硬件）地址。BOOT-P 是用来确定网络上的哪个装置没有 IP 地址并将您所选择的装置分配 IP 地址。Ping 用于检查具有指定 IP 地址的装置与主计算机之间的通信。

DMC-2100/2200 能够通过可发送 TCP/IP 或 UDP/IP 邮袋的任何应用软件与主计算机进行通信。一个很好的例子是 Telnet，它是与大多数 Windows 系统捆绑的工具软件。

4. 数据纪录

DMC-2×00 通过使用单个命令 QR 就能够提供许多状态信息。此命令与 QZ 命令配合使用对于存取整个控制器状态非常有用。QR 命令会返回 4 字节头信息和通过命令参数所指定的指定块信息如：QR ABCDEFGHST。各个参数按照如下数据纪录表对相应的块信息。如果不给出参数，就返回整个数据纪录表。注意：数据纪录空间大小取决于轴数。

4. 1 数据纪录

数据类型	内容分类	块
UB	1st byte of header	Header
UB	2nd byte of header	Header
UB	3rd byte of header	Header
UB	4rth byte of header	Header
UW	sample number	I block
UB	general input 0	I block
UB	general input 1	I block
UB	general in put 2	I block
UB	general input 3	I block
UB	general input 4	I block
UB	general input 5	I block
UB	general input 6	I block
UB	general input 7	I block
UB	general input 8	I block
UB	general input 9	I block
UB	general output 0	I block
UB	general output 1	I block
UB	general output 2	I block
UB	general output 3	I block
UB	general output 4	I block
UB	general output 5	I block
UB	general output 6	I block
UB	general output 7	I block
UB	general output 8	I block
UB	general output 9	I block
UB	error code	I block

UB	general status	I block
UW	segment count of coordinated move for S plane	S block
UW	coordinated move status for S plane	S block
SL	distance traveled in coordinated move for S plane	S block
UW	segment count of coordinated move for T plane	T block
UW	coordinated move status for T plane	T block
SL	distance traveled in coordinated move for T plane	T block
UW	x,a axis status	A block
UB	x,a axis switches	A block
UB	x,a axis stopcode	A block
SL	x,a axis reference position	A block
SL	x,a axis motor position	A block
SL	x,a axis position error	A block
SL	x,a axis auxiliary position	A block
SL	x,a axis velocity	A block
SW	x,a axis torque	A block
SW	x,a axis analog	A block
UW	y,b axis status	B block
UB	y,b axis switches	B block
UB	y,b axis stopcode	B block
SL	y,b axis reference position	B block
SL	y,b axis motor position	B block
SL	y,b axis position error	B block
SL	y,b axis auxiliary position	B block
SL	y,b axis velocity	B block
SW	y,b axis torque	B block
SW	y,b axis analog	B block
UW	z,c axis status	C block
UB	z,c axis switches	C block
UB	z,c axis stopcode	C block
SL	z,c axis reference position	C block
SL	z,c axis motor position	C block
SL	z,c axis position error	C block
SL	z,c axis auxiliary position	C block
SL	z,c axis velocity	C block
SW	z,c axis torque	C block
SW	z,c axis analog	C block
UW	w,d axis status	D block
UB	w,d axis switches	D block
UB	w,d axis stopcode	D block
SL	w,d axis reference position	D block
SL	w,d axis motor position	D block
SL	w,d axis position error	D block
SL	w,d axis auxiliary position	D block

SL	w,d axis velocity	D block
SW	w,d axis torque	D block
SW	w,d axis analog	D block
UW	e axis status	E block
UB	e axis switches	E block
UB	e axis stopcode	E block
SL	e axis reference position	E block
SL	e axis motor position	E block
SL	e axis position error	E block
SL	e axis auxiliary position	E block
SL	e axis velocity	E block
SW	e axis torque	E block
SW	e axis analog	E block
UW	f axis status	F block
UB	f axis switches	F block
UB	f axis stopcode	F block
SL	f axis reference position	F block
SL	f axis motor position	F block
SL	f axis position error	F block
SL	f axis auxiliary position	F block
SL	f axis velocity	F block
SW	f axis torque	F block
SW	f axis analog	F block
UW	g axis status	G block
UB	g axis switches	G block
UB	g axis stopcode	G block
SL	g axis reference position	G block
SL	g axis motor position	G block
SL	g axis position error	G block
SL	g axis auxiliary position	G block
SL	g axis velocity	G block
SW	g axis torque	G block
SW	g axis analog	G block
UW	h axis status	H block
UB	h axis switches	H block
UB	h axis stopcode	H block
SL	h axis reference position	H block
SL	h axis motor position	H block
SL	h axis position error	H block
SL	h axis auxiliary position	H block
SL	h axis velocity	H block
SW	h axis torque	H block
SW	h axis analog	H block

NOTE: UB = Unsigned Byte, UW = Unsigned Word, SW = Signed Word, SL = Signed Long Word

4. 2 状态信息和轴开关信息说明:

Header Information - Byte 0, 1 of Header:

BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
1	N/A	N/A	N/A	N/A	I Block Present in Data Record	T Block Present in Data Record	S Block Present in Data Record
BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
H Block Present in Data Record	G Block Present in Data Record	F Block Present in Data Record	E Block Present in Data Record	D Block Present in Data Record	C Block Present in Data Record	A Block Present in Data Record	B Block Present in Data Record

Bytes 2, 3 of Header:

Bytes 2 and 3 make a word which represents the Number of bytes in the data record, including the header.

Byte 2 is the low byte and byte 3 is the high byte

NOTE: The header information of the data records is formatted in little endian.

General Status Information (1 Byte)

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
Program Running	N/A	N/A	N/A	N/A	Waiting for input from IN command	Trace On	Echo On

Axis Switch Information (1 Byte)

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
Latch Occurred	State of Latch Input	N/A	N/A	State of Forward Limit	State of Reverse Limit	State of Home Input	SM Jumper Installed

Axis Status Information (2 Byte)

BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
Move in Progress	Mode of Motion PA or PR	Mode of Motion PA only	(FE) Find Edge in Progress	Home (HM) in Progress	1st Phase of HM complete	2 nd Phase of HM complete or FI command issued	Mode of Motion Coord. Motion
BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
Negative Direction Move	Mode of Motion Contour	Motion is slewing	Motion is stopping due to ST or Limit Switch	Motion is making final decel.	Latch is armed	Off-On-Error occurred	Motor Off

Coordinated Motion Status Information for S or T plane (2 Byte)

BIT 15	BIT 14	BIT 13	BIT 12	BIT 11	BIT 10	BIT 9	BIT 8
Move in Progress	N/A	N/A	N/A	N/A	N/A	N/A	N/A
BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
N/A	N/A	Motion is slewing	Motion is stopping due to ST or Limit Switch	Motion is making final decel.	N/A	N/A	N/A

4. 3 关于速度、转矩信息的注意事项:

在数据纪录中返回的速度信息是使用 TV 命令所返回值的 64 倍。请参见第二篇命令手册中相关 TV 命令的说明。

转矩信息表达为 ± 32767 范围内的数。最大反向转矩为 -32767。最大正向转矩为 32767。零转矩为 0。

4. 4 QZ 命令

当使用 QR 命令时, QZ 命令就非常有用, 这是因为 QZ 命令提供了有关控制器和数据纪录方面的信息。QZ 命令返回如下 4 字节信息:

字节号	信 息
0	轴数
1	通用数据纪录块中的字节数
2	联动控制平面数据纪录块中的字节数
3	各轴数据纪录块中的字节数

5. 控制器对命令的响应

大多数 DMC-2x00 指令用跟随有参数的两个字符来表达，各指令必须用回车或分号来结束。

指令以 ASCII 码发送，DMC-2x00 对各 ASCII 字符（一个字一个字）逐一进行译码。控制器对每条指令译码大约占用 0.5msec 时间。不过，由于使用 FIFO 缓存，因此，PC 机能以更快的速率给控制器发送数据。

在对指令进行译码之后，DMC-2x00 回复一个响应给发送过命令的口。如果命令是有效的话，控制器就返回一个 (:)，而当命令是无效的时，则返回一个 (?)。例如：控制器将响应由 USB 口发送的命令，就由 USB 口回复，由 RS232 口发送的命令回复到 RS232 口，而由 Ethernet 口发送的命令回复到 Ethernet 口。

对于返回数据的那些指令，如 TP，DMC-2x00 就会弹回由回车符，进行和冒号所跟随的数据。

在发送各命令之后，以防出错，用冒号加以确认就是一个很实际的应用。控制器提供回响功能就能使 DMC-2x00 与发送的数据响应相协调。通过发送命令 EO1 给控制器，就使回响功能有效。

6. 由控制器产生的非请求信息

控制器正在执行程序时，可以产生由 USB 口 (DMC-2000)，主 RS232 口或 Ethernet (DMC-2100/2200) 发送响应。使用 MG 或 IN 命令时，此响应的结果是读回字符串信息或者是命令错误。由于这些响应不是对命令的直接响应，因此，称其为非请求信息，或一般信息。

用指定口参数能将信息引向指定口，参见第二篇命令手册中关于 MG 和 IN 命令的描述。如果未明确指定口，那么就会将这些非请求信息发送到缺省口。缺省口由系统复位时 USB/Ethernet 指拨开关的状态所决定。

控制器有一个特殊命令 CW，它能够影响非请求信息的格式。此命令由 GALIL 软件使用，来把响应与命令行和非请求信息相区别。命令 CW1 使控制器把 ASCII 字符的高位设置成所有非请求字符 1。这样就可能引起字符对有些终端来说出现混肴。用命令 CW2 使此功能无效。有关详细说明，参见第二篇命令手册中的 CW 命令。

当使用握手时（硬件或软件握手），由控制器产生的字符在从控制器出发之前，放在 FIFO 缓冲区中。缓冲区的大小是 64 字节，RS232 缓冲区的大小是 128 字节。当此缓冲区放满时，控制器必须停止执行命令或者忽略掉对输出产生的字符。命令 CW，1 使控制器在 FIFO 放满期间忽略掉来自控制器的所有输出。CW，0 使控制器停止执行命令，直至获得更多的 FIFO 空间为止。当使用硬件软件握手时，此命令很有用，此时，将会暂时中断控制器与终端之间的通信连线。在此情况下，字符将继续堆入控制器，直到 FIFO 充满为止。详见第二篇命令手册中 CW 命令说明。

7. GALIL 软件工具和库文件

API（应用编程接口）软件由 GALIL 向广大用户提供。API 软件用 C 写成，包含在 GALIL CD-ROM 中，用户能够在 DOC 和 WINDOWS 环境下（16 位和 32 位 WINDOWS）用 API 进行二次应用开发。用 API 软件，用户能够把已存在的库函数直接合并到 C 程序中。

GALIL 也开发了 Activex Tool Kit 套件，提供 VBX，16bitOCX 和 32bitOCX 来处理包括支持中断在内的所有通信，这些目标码直接装到 VB Labview C 或接受 Activex 控件的任何软件包中并作为实时控制软件的一部分，有关涉及高级编程应用开发的详细内容，请参考相关 CD-ROM 或中宝伦公司联系。

第五章 命令基础

1. 说明

在了解了运动控制器系统的构成、硬件连接、通信之后，本章将对运动控制器的命令作一些基础性的介绍，以便用户在后续章节中能够深入领会运动编程及应用开发中所涉及到的命令。

DMC-2x00 为用户提供了 100 多条命令用于运动控制和机械参数设置。这些命令涉及到初始化、查询状态、配置数字滤波器等方面，而且这些命令能以 ASCII 码或二进制码发送。

以 ASCII 码编写运动程序，这些指令就类似于 BASIC，使用起来非常容易。指令由两个大写字母组成，在发音上与相应的功能相对应，特别易记，并可联想。例如：指令 BG 指开始运动 (Begin Motion)，ST 指停止运动 (Stop)；以二进制码方式，命令用 80~FF 之间的二进制码表示。

用户能够实时通过总线（通信口）发送 ASCII 命令，由 DMC-2x00 控制器立即执行，或者将整个一组命令下载到 DMC-2x00 存储器中，再执行。将命令编写成程序以备执行，这方面的内容请参考“应用编程”一章。在应用编程中不能使用二进制命令。

本章重点描述 DMC-2x00 指令集和语句。命令总集及 DMC-2x00 全部指令列表请参见第二篇命令手册中相关章节。

2. 命令语法——ASCII 码

DMC-2x00 指令用两个 ASCII 大写字符后跟可用参数来表达，可以在指令和参数之间插入空格，用分号或回车来终止指令。

注意：如果您使用 GALIL 终端工具软件来编写程序，只有在给出《return》回车命令后，才会处理这些命令。这样就允许用户按单行把许多命令分开，直到用户给了《return》命令，才开始执行。

重要提示：所有 DMC-2x00 命令均以大写字符发送。

例如，命令

PR 4000 《return》 相对位置

PR 是用于指定相对位移量的两字符指令，4000 是以计数单位表示所要运动的距离。《return》使该条指令结束。PR 和 4000 之间的空格为可选择（可有也可无）。

为了对 A, B, C, D 轴指定数据，就用逗号将轴分开，如果对一个轴没有指定数据，但仍然需要逗号，如下例所示，在此情况下，就维持先前的值。

要想查看各命令的当前值，就对所提问的各轴用“？”跟随在命令之后。

PR 1000	只对 A 轴指定 1000
PR , 2000	只对 B 轴指定 2000
PR ,, 3000	只对 C 轴指定 3000
PR ,,, 4000	只对 D 轴指定 4000
PR 2000, 4000, 6000, 8000	对 A, B, C, D 轴指定位移值
PR , 8000,, 9000	只对 B, D 轴指定位移值
PR ? , ? , ? , ?	查询 A, B, C, D 轴位移值
PR , ?	只查询 B 轴位移值

DMC-2x00 提供了好几种方法对各轴指定数据，下面是用单个轴指定器（如 A, B, C, D 等）单独指定数据的举例。用等号来对该轴分配数据。

举例如下：

PRA=1000	对 A 轴指定相对位移量 1000
ACB=200000	对 B 轴指定加速度 200000

与分配数据不同，有些命令会要求一个轴或多个轴动作。例如，ST AB 就使 A 轴和 B 轴停止运动，在此情况下，由于用相对应的字母 A,B,C,D 指定相应轴，因此就不需要逗号。如果没有参数跟在指令后面，那么就会使所有轴发生动作。以下是有关这方面的举例：

BG A	只启动 A 轴
BG B	只启动 B 轴
BG A B C D	启动所有轴
BG B D	只启动 B D 轴
BG	启动所有轴
BG A B C D E F G H	启动所有 8 个轴
BG D	只启动 D 轴

一轴以上的联动控制

当要求联动控制运动时，用字母 S 和 T 来指定联动平面。例如：

BG S	开始联动程序，在 S 坐标系
BG TW	开始联动程序，在 T 坐标系，命令 W 轴运动。

3. 命令语法——二进制码

有些命令有等效的二进制数，通信方式的执行速度比 ASCII 码命令快得多。二进制格式只能用于从 PC 机发送命令时，他不能嵌入到应用程序中。

3. 1 二进制命令格式

所有二进制命令都有 4 字节头，并有数据域跟随。4 个字节以十六进制格式指定。

头格式：

字节 1：指定 80~FF 之间的命令代码，完整的二进制命令代码表列表如下。

字节 2：指定各个域中的字节号，0，1，2，4，6，如下：

00	非数据域（例如 SH 或 BG）
01	每域 1 个字节
02	一个字（每域 2 字节）
04	每域一个长字（4 字节）
06	GALIL 实际格式（整数和小数）

字节 3：指定命令是否适用于联动控制运动，如下：

00	非联动控制运动
01	联动控制运动

例如，命令 ST S 使矢量运动停止，等效二进制命令的第 3 字节为 01。

字节 4：指定轴号或数据域，如下：

Bit7	= H 轴或第 8 数据域
Bit6	= G 轴或第 7 数据域
Bit5	= F 轴或第 6 数据域
Bit4	= E 轴或第 5 数据域
Bit3	= D 轴或第 4 数据域
Bit2	= C 轴或第 3 数据域
Bit1	= B 轴或第 2 数据域
Bit0	= A 轴或第 1 数据域

3. 2 数据域格式

数据域必须与格式字节和轴字节相一致。例如，

命令 PR1000,-500 为: A7 02 00 05 03 E8 FE 0E

其中: A7 是 PR 的二进制码
 02 指定各数据域为 2 字节
 00 S 对 PR 无效
 05 指定 bit0 对 A 轴有效, bit2 对 C 轴有效 ($2^0+2^2=5$)
 03 H 表示 1000
 FE 0E 表示-500

举例:

命令 ST ABCS 为:

A1 00 01 07

其中: A1 是 ST 命令的二进制码
 00 指定 0 数据域
 01 指定停止联动控制轴 S
 07 表示停止 X(bit0),Y(bit1)和 Z (bit2) 轴, $2^0+2^1+2^2=7$

3. 3 二进制命令表

COMMAND	NO.	COMMAND	NO.	COMMAND	No.
reserved	80	reserved	ab	reserved	d6
KP	81	reserved	ac	reserved	d7
KI	82	reserved	ad	RP	d8
KD	83	reserved	ae	TP	d9
DV	84	reserved	af	TE	da
AF	85	LM	b0	TD	db
KF	86	LI	b1	TV	dc
PL	87	VP	b2	RL	dd
ER	88	CR	a3	TT	de
IL	89	TN	b4	TS	df
TL	8a	LE, VE	b5	TI	e0
MT	8b	VT	b6	SC	e1
CE	8c	VA	b7	reserved	e2
OE	8d	VD	b8	reserved	e3
FL	8e	VS	b9	reserved	e4
BL	8f	VR	ba	TM	e5

AC	90	reserved	bb	CN	e6
DC	91	reserved	bc	LZ	e7
SP	92	CM	bd	OP	e8
IT	93	CD	be	OB	e9
FA	94	DT	bf	SB	ea
FV	95	ET	c0	CB	eb
GR	96	EM	c1	II	ec
DP	97	EP	c2	EI	ed
DE	98	EG	c3	AL	ee
OF	99	EB	c4	reserved	ef
GM	9a	EQ	c5	reserved	f0
reserved	9b	EC	c6	reserved	f1
reserved	9c	reserved	c7	reserved	f2
reserved	9d	AM	c8	reserved	f3
reserved	9e	MC	c9	reserved	f4
reserved	9f	TW	ca	reserved	f5
BG	a0	MF	cb	reserved	f6
ST	a1	MR	cc	reserved	f7
AB	a2	AD	cd	reserved	f8
HM	a3	AP	ce	reserved	f9
FE	a4	AR	cf	reserved	fa
FI	a5	AS	d0	reserved	fb
PA	a6	AI	d1	reserved	fc
PR	a7	AT	d2	reserved	fd
JG	a8	WT	d3	reserved	fe
MO	a9	WC	d4	reserved	ff
SH	aa	reserved	d5		

第六章 运动编程

1. 概述

DMC-2×00 提供了好几种运动方式，它包括定位控制、手动、联动控制运动、电子凸轮运动、电子齿轮等。用户可以非常方便的利用这些现有的运动方式根据控制对象编写出应用程序。本章将对这些运动方式逐个进行讨论。

DMC-2×10 是单轴控制器，只使用 A 轴运动。以此类推，DMC-2×20 使用 A, B 轴，DMC-2×30 使用 A, B, C 轴，DMC-2×40 使用 A, B, C, D 轴，DMC-2×50 使用 A, B, C, D, E 轴，DMC-2×60 使用 A, B, C, D, E, F 轴，DMC-2×70 使用 A, B, C, D, E, F, G 轴，DMC-2×80 使用 A, B, C, D, E, F, G, H 轴。

以下举一些例子以帮助您进一步理解有关的运动方式。

举 例 应 用	运 动 方 式	命 令
绝对或相对定位，其中各轴独立运动，运动轮廓为梯形包络线	独立轴定位	PA,PR , SP,AC,DC
速度控制，这里，未预设终点，运动由停止命令来停止	独立运动	JG, AC,DC

		ST
描述为增量位置对应时间的运动轨迹	轮廓运动	CM,CD,DT,WC
2, 3, 4 轴联动控制运动, 这里轨迹由直线段来描述	直线插补运动	LM,LI,LE VS,VR,VA,VD
由圆弧和直线段构成的 2 维运动轨迹, 例如雕刻或织挑、织补	联动运动	VM,VP,CR VS,VR,VA,VD VE
第 3 轴必须与 2 维运动轨迹保持正切, 例如: 刀具切割。	带有指定正切轴的联动运动	VM,VP,CR, VS,VA,VD TN,VE
电子齿轮同步运动, 其中, 从动轴与主动轴成比例运动, 能够以两个方向运动。	电子齿轮同步运动	GA,GR GM(若是龙门同步)
主/从运动, 这里, 从动轴必须跟随主动轴, 例如: 输送带速度	电子齿轮同步运动	GA GR
按任意轮廓或数字模型所预置轮廓运动, 例如, 正弦, 余弦, 梯形轮廓	轮廓方式	CM,CD DT,WC
示教, 记录和录返	具有自动数组捕获功能的轮廓方式	CM,CD,DT,WC RA,RD,RC
间隙补偿	双位置闭环	DV
基于主动轴编码器位置的轮廓跟踪运动	电子凸轮运动	EA,EM,EP,ET EB,EG,EQ
在以独立轴定位方式工作的平滑运动	独立运动平滑处理	IT
在以矢量或直线插补定位方式工作时的平滑运动	矢量平滑	VT
以步进电机运行时的平滑运动	步进电机平滑处理	KS
龙门驱动—两个轴由龙门架耦合	龙门同步驱动方式	GR GM

2. 独立轴定位控制

所谓独立轴定位控制, 就是说各轴之间的运动在数学上相互独立, 为非耦合(或联动)关系, 各轴按照程序中所规定的运动轨迹完成各自的定位控制。用户为各轴指定目标绝对位置(PA)或相对位置(PR), 速度(SP), 加速度(AC), 减速度(DC), 让其开始运动(BG), DMC-2×00 控制器就自动产生相应的速度和位置轨迹, 完成定位控制运动。

注意: 当控制器轮廓分配完成时, 由于电机及负载惯性, 实际电机运动并不一定立即完成, 但这并不影响发送下一条运动命令。

控制器可以对所有各轴同时或分别发送 BG 命令, 需要用 A, B, C, D 来选择所要运动的轴。当未指定轴号时, 就意味着使所有轴开始运动。在运动期间, 可以随时改变运动速度(SP)和加速度(AC), 不过, 直到运动完成后, 才可以改变减速度(DC)和位置值(PR 或 PA)。请记住, 轮廓分配结束时, 运动就完成, 而不是在实际电机处于定位时。在到达终点位置之前, 可以随时发送停止命令(ST)使电机减速停止。

只要附加运动与正在执行的运动方向相同，在运动期间就可以指定增量位置(IP)，此时用户只指定新的位置增量 n ，新的目标位置就等于原有目标位置+增量 n 。一旦接收到 IP 命令，就会产生新轮廓，使运动达到新的终点位置。IP 命令不需要 BG。

注意：如果电机不在运动中，IP 命令就等效于 PR 和 BG 命令的组合。

2. 1 独立轴定位命令汇总

命 令	说 明
PR A,B,C,D	指定相对距离
PA A,B,C,D	指定绝对位置
SP A,B,C,D	指定运动速度
AC A,B,C,D	指定加速度
DC A,B,C,D	指定减速度
BG ABCD	开始运动
ST ABCD	在到达运动终点之前停止
IP A,B,C,D	更改位置目标值
IT A,B,C,D	独立运动平滑时间常数 (S 曲线)
AM ABCD	轮廓分配完成用条件启动
MC ABCD	定位完成用条件启动


DMC-2×00 也允许使用单轴指定，如 PRB=2000

2、2 独立轴定位操作数汇总

命 令	说 明
- ACx	读取由 X 所指定轴的加速度
- DCx	读取由 X 所指定轴的减速度
- SPx	读取由 X 所指定轴的速度
- PAx	若“X”轴正在运动中，读取当前实际位置值，而如果处于运动状态，则读取当前命令位置。
- PRx	读取由“X”所指定轴的当前增量位置值

2. 3 举例：

1) 绝对位置运动

指令	说 明
 PA 10000, 20000	指定 A, B 轴绝对位置
AC 1000000, 1000000	指定 A, B 轴加速度
DC 1000000, 1000000	指定 A, B 轴减速度
SP 50000, 30000	指定 A, B 轴速度
BG AB	开始运动


2) 多个运动顺序

要求运动参数：

A 轴：	2000 计数单位	位置
	15000 计数单位/sec	速度
	500000 计数单位/sec ²	加速度
B 轴：	1500 计数单位	位置
	10000 计数单位/sec	速度

	500000 计数单位/sec ²	加速度
C 轴:	100 计数单位	位置
	5000 计数单位/sec	速度
	500000 计数单位/sec ²	加速度

此例要求 A, B, C 轴做相对位置运动, 各轴运动间隔为 20msec, 图 6.1 表示轴的速度轮廓。此例的程序如下:

指令	说明
# A	程序标号
 PR2000,1500,100	A,B,C 轴运动距离分别为 2000, 1500, 100 计数单位
SP15000,10000,5000	A,B,C 轴运动速度分别为 15000, 10000, 5000 计数单位/sec
AC500000,500000,500000	各轴加速度为 500000 计数单位/sec ²
DC500000,500000,500000	各轴减速度为 500000 计数单位/sec ²
BGA	开始 A 轴运动
WT 20	等待 20msec
BGB	开始 B 轴运动
WT20	等待 20 msec
BGC	开始 C 轴运动
EN	程序结束

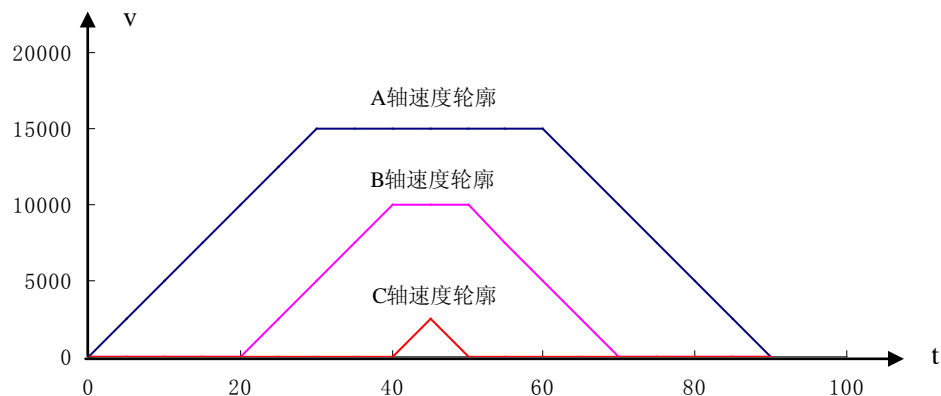


图 6.1 A BC 速度轮廓

图注: A, B 两轴有一个“梯形”速度包络线, 而 C 轴有一个“三角形”速度包络线。A 轴加速度到指定速度, 以恒速运动。然后减速到达命令位置所指定的终点。C 轴加速, 但在到达指定速度之前, 就必须减速以使该轴停在指定的位置。所有三个轴均有相同的加, 减速度, 因此, 三个速度轮廓的上升, 下降沿的斜率是相同的。

3. 独立 JOG 运动

JOG 运动方式非常灵活, 这是因为在运动期间, 可以随时变更运动速度、加速度和运动方向。用户对各轴规定 **JOG 速度(JG)**, 加速度(AC)和减速度 (DC)。运动方向由 JG 参数的符号来决定; 当用 BG 使运动开始时, 电机加速到达所设速度, 并以此速度连续运转, 直到有新的速度给定或发送 ST 命令。如果在运动期间改变 JOG 速度, 控制器就会使电机加速或减速达到新速度。

通过使用 IP 命令能够使电机位置发生瞬时改变, 一接收到此命令, 控制器就命令电机到达给

定增量加上当前位置的新目标位置，此命令对于在电机运行过程中，想使两个电机的位置同步是非常有用的。

注意：在以 JOG 方式运行期间，控制器以位置闭环方式工作。DMC-2×00 控制器将速度轮廓转换成位置包络线，每个采样周期产生新的位置目标值。这种控制方法用精确锁相进行精密速度调节。

1). JOG 指令汇总

指 令	说 明
AC A,B,C,D	指定加速度
BG ABCD	开始运动
DC A,B,C,D	指定减速度
IP A,B,C,D	瞬时指定增量位置
IT A,B,C,D	独立运动平滑处理时间常数
JG ±A,B,C,D	指定 JOG 速度和方向
ST ABCD	停止运动

可以单独对运动轴设置参数，如 JGB=2000（对 B 轴设定 JOG 速度 2000）

2). 独立运动轴操作数汇总

操 作 数	说 明
-ACx	读取由“X”所指定的加速度
-DCx	读取由“X”所指定的减速度
-SPx	读取由“X”所指定的 JOG 速度
-TVx	读取由“X”所指定的实际速度（0.25sec 的均值）

3). 举例：

例1 只有 X 轴以 JOG 方式运动

使 A 轴以 50000 计数单位/sec 速度做 JOG 运动，在 A 轴电机到达 JOG 速度后，使 C 轴以 25000 计数单位/sec 速度做反方向运动。

指 令	说 明
# A	程序标号
AC20000,,20000	指定 A,C 轴加速度为 20000 计数单位/sec ²
DC20000,,20000	指定 A,C 轴减速度为 20000 计数单位/sec ²
JG50000,-25000	指定 A,C 轴 JOG 速度和方向
BGA	开始 A 轴运动
ASA	等待 A 轴速度到达
BGC	开始 C 轴运动
EN	程序结束

例2 Joystick 手操杆运动

JOG 速度也能够用模拟输入(例如手操杆)加以改变，假设 IOV 输入，对应速度必须是 50000 cts/sec。

指 令	说 明
# JOY	程序标号

JGO

BGA

B

V1=@AN[1]

Vel=V1*50000/10

JG Vel



JP# B

设置 JOG 方式

开始运动

循环程序标号

读取模拟口 1 输入

计算速度

改变 JG 速度

循环

4. 直线插补运动

DMC-2×00 提供 8 轴直线插补功能。在直线插补方式中，各轴沿规定的轨迹以矢量速度，加速度，减速度联动运动；运动轨迹根据各轴的增量距离来产生。在连续插补运动中，可以给定无限增量线段，使直线插补方式完全跟随工件运动轨迹，对总的运动长度没有限制。

LM 命令为插补选择直线插补方式和插补轴。例如，LM BC 只选择 B 轴和 C 轴为直线插补。使用直线插补方式时，如果不补改变直线插补轴，那么只需要指定一次 LM 命令。

4. 1 指定联动平面

对于直线插补方式或矢量方式来讲，DMC-2×00 考虑到两套联动轴，他们分别用字母 S 和 T 来分辨。

要想规定矢量命令，必须先辨别联动平面。用 CAS 命令来辨别 S 平面，而用 CAT 命令来辨别 T 平面。如果未用 CA 命令改变坐标系，那么所有矢量命令都适用于当前有效坐标系。

4. 2 指定直线段

用命令 LI,a,b,c,d,e,f,g,h 来为各轴指定增量运动距离，这就意味着以当前的轴位置值，规划运动。在 BGS 命令之前，可以给出正 511 个增量线段。一旦开始运动，还可以给控制器发送附加 LI 线段。

在运动启动之前，能用清除线段命令 CS 来删除存储在缓冲区中的 LI 线段。要想停止运动，就使用指令 STS 或 AB。ST 命令使电机减速停止，而命令 AB 使电机即刻停止并使程序终止执行，命令 AB1 只使运动终止。

必须使用直线插补结束命令 LE 来指定直线插补运动程序结束，此命令告诉控制器减速停止在最后一条 LI 命令之后。如果未给定 LE 命令，就必须用终止命令 AB1 来终止运动程序。

用户的责任在于在 DMC-2×00 程序缓冲区中保持足够的 LI 线段以确保连续运动。如果控制器未接收到附加 LI 线段和 LE 命令，在最后一个矢量线段完成之后，控制器就会立即停止运动，也没有可控制的减速度。LM? 或-LM 读取能够发送到缓冲区的 LI 线段的可用空间。返回值为 511 意味着缓冲区是空的，可以发送 511 个 LI 线段。若返回值为 0，则意味着缓冲区是满的，不能再发送附加 LI 线段。只要缓冲区未满，就能够以 PC 机总线速度发送附加 LI 线段。

指令-CS 读取线段计数器值，当处理线段时，-CS 增加，此计数器值以 0 开始计数。此功能使得主计算机能够确定正在处理哪条线段。

4. 3 附加命令

用命令 VSn,VAn 和 VDn 来指定矢量速度，矢量加速度和减速度，DMC-2×00 以 LM 方式中所指定的轴为基础计算矢量速度。例如，LM ABC 指定 A,B,C 轴为直线插补轴，此例中的矢量速度用下式进行计算：

$$V_s S^2 = A S^2 + B S^2 + C S^2$$

其中：AS,BS,CS 分别为 A,B,C 轴的速度。

控制器总是使用由 LM 所指定的轴定义来计算速度，而不是 LI。

用 VT 来设置联动控制轴 S 曲线平滑参数；命令 AVn 是“矢量运动完成后”条件启动命令，它使程序暂停执行,直到矢量距离 n 到达之后,才重新启动执行后续程序。

4. 3. 1 对各插补线段指定矢量速度

指令 VS 具有立即效应，因此，必须在需要时刻给定。在某些应用中，例如 CNC,需要对不同的运动线段加附各种不同的速度；这可以用两个函数来进行 (<n 和>m)。




例如：LI a,b,c,d <n>m

第一个命令<n 等效于在给定线段的起点命令 VS_n，在其它约束下，使运动朝着新的速度加速；

第二个函数>m，需要矢量速度在线段终点要达到的值 m。注意：函数>m 可在给定的线段内或在前一线段期间开始减速，在给定的 VA,VD 值条件下，满足最终速度需要。

不过，请注意：控制器一次只有一个>m 命令起作用，因此，一个函数可以用另一个加以屏蔽。例如，如果函数.>100000 由>5000 跟随其后，且减速距离不够，那么就不会满足第二个条件；控制器试图将速度降到 5000，但会到达与此不同的点。

作为一个例子，看看以下程序。

指 令	说 明
# ALT	程序标号
DP0 ,0	定义 A,B 轴位置为 0
 LM AB	定义 A,B 轴之间为直线插补方式
LI 4000,0<4000>1000	指定第一个直线插补线段 A 轴走 4000 单位 B 轴走 0 单位， 矢量速度为 4000，结束速度为 1000，
LI 1000,1000<4000>1000	指定第二个直线插补线段 A 轴走 1000 单位 B 轴走 1000 单位，矢量速度为 4000，结束速度为 1000，
 LI 0,5000<4000>1000	指定第三个直线插补线段 A 轴走 0 单位 B 轴走 5000 单位， 矢量速度为 4000，结束速度为 1000，
 LE	结束直线插补线段
BG S	开始运动
EN	程序结束

4. 3. 2 改变进给率

命令 VR_n 使进给率 VS 以 0.0001 的分辨率在 0~10 之间成比例变化。此命令立即起作用并使 VS 按比例变化。VR 也适用于用“<”指定矢量速度时。此命令对于进给率修调功能非常有用。VR 不对加速度进行比例处理。例如，VR0.5 使 VS2000 减小一半。

4. 4 直线插补命令汇总

命 令	说 明
LM abcdefgh	指定直线插补轴
LM?	读取控制器缓冲区中直线插补线段可用空间，0 表示缓冲区已满，512 表示缓冲区空
LI a,b,c,d,e,f,g,h<n	指定相对于当前位置的增量距离，并分配矢量速度 n。
VS _n	指定矢量速度
VA _n	指定矢量加速度
VD _n	指定矢量减速度
VR _n	指定矢量速度比率
BGS	开始直线插补程序
CS	清除缓冲区中程序段
LE	直线插补结束——位于 LI 命令的最后

LE?	读取矢量长度（在 2147483647 之后复位）
AMS	在矢量运动完成之后的条件启动
AV _n	在相对矢量距离 <i>n</i> 到达之后的条件启动
VT	矢量运动 S 曲线平滑常数

4. 5 直线插补操作数汇总

操 作 数	说 明
-AV	读取已运动的相对矢量距离
-CS	插补线段计数器——读取程序中的插补线段数，以 0 开始
-LE	读取矢量长度（在 2147483647 之后复位）
-LM	读取控制器缓冲区中直线插补线段可用空间，0 代表缓冲区已满，511 表示缓冲区空
-VP _m	读取沿轨迹最后一个数据点的绝对坐标值，（ <i>m</i> =A,B,C,D,E,F,G,H）

为了表明查询运动状态的功能，考虑一下举例#LMOVE 中的第一个运动线段，其中，A 轴朝点 A=5000 运动。假设，当 A=3000 时，用命令“MG-AV”查询控制器，返回的值将是 3000，-CS,-VPA,-VPB 的值将为 0。现在假设：在第二个插补线段 B=2000 时重复查询，在此点的-AV 值是 7000，-CS 等于 1，-VPA=5000,-VPB=0。

4. 6 举例：

例 1 直线插补运动

在本例中，要求 A，B 轴进行 90° 转向，为了在拐角周围减慢速度，我们使用 AV4000 条件启动命令，将速度减慢到 1000cts/sec,一旦电机到达拐角，速度就增回到 4000cts/sec。

指 令	说 明
# LMOVE	程序标号
DP0,0	定义 A B 轴位置为 0
LMAB	定义 A B 轴为直线插补关系
LI5000,0	指定第一条直线插补线段
LI0,5000	指定第二条直线插补线段
LE	直线插补结束
VS4000	指定矢量速度
BGS	开始运动
AV4000	设置条件启动，等待到达矢量距离 4000
VS1000	改变矢量速度
AV5000	设置条件启动，等待到达矢量距离 5000
VS4000	改变矢量速度
EN	程序结束

例 2 直线插补运动

在 CD 平面（即 ZW 平面）做联动控制直线插补运动，以 100000cts/sec 矢量速度和 100000cts/sec² 的矢量加速度移动到坐标点 40000，30000cts。

指 令	说 明
LM CD	指定直线插补轴
LI , 40000,30000	指定 C,D 距离
LE	直线插补结束

VS 100000 指定矢量速度
 VA 1000000 指定矢量加速度
 VD 1000000 指定矢量减速度
 BG S 开始运动

注意：上述程序指定的矢量速度 VS,并不是实际轴速度 VC 和 VD，轴速度由下式求得：

$$VS = \sqrt{VC^2 + VD^2}$$

各轮廓曲线见图 6.2

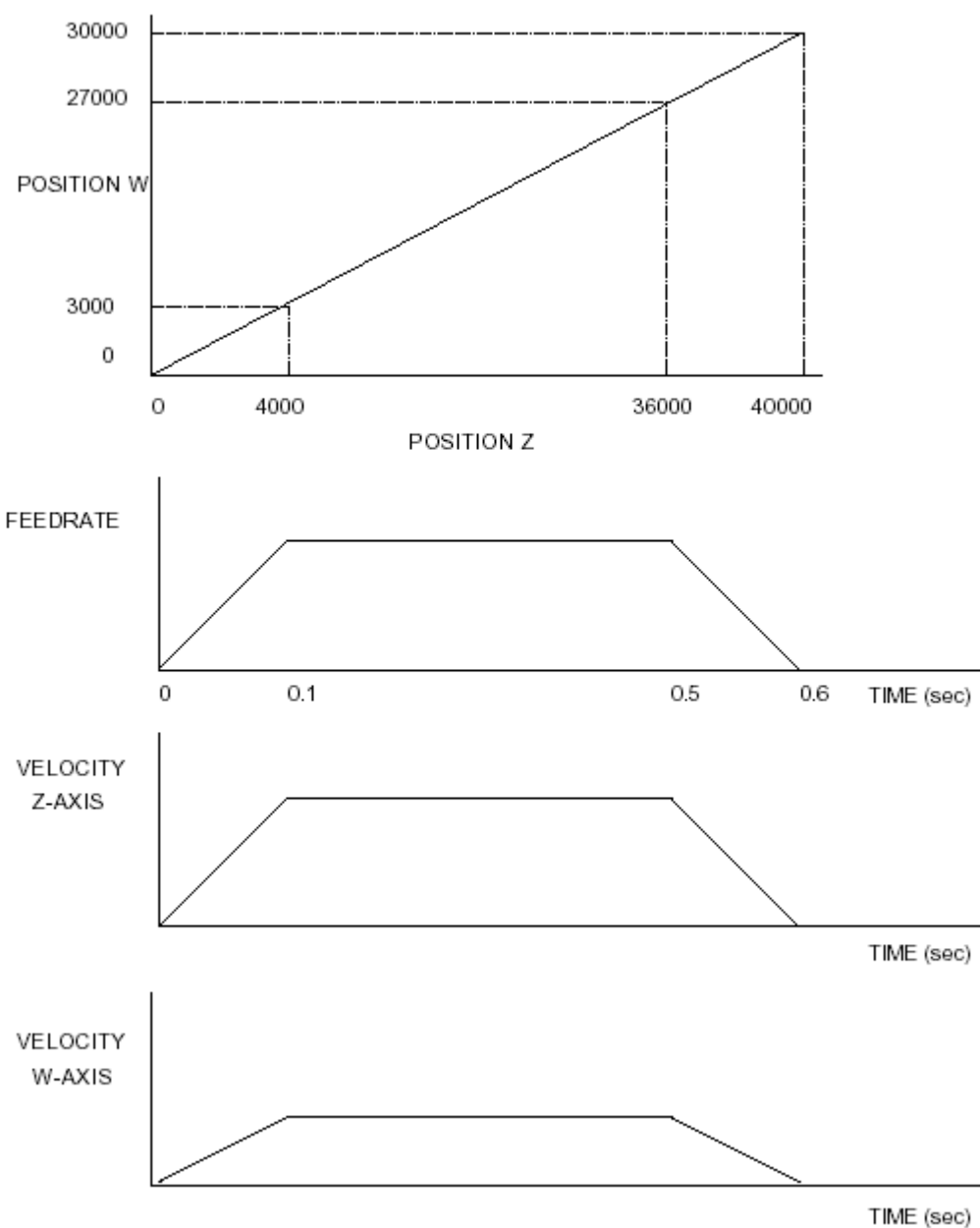


图6.2 – 直线插补

例3 多重循环直线插补运动

本例是 AB 平面 (XY 平面) 的联动直线插补运动, 使用数组 VA 和 VB 来存储由程序 #LOAD 所填入的 750 个增量距离。

指 令	说 明
# LOAD	LOAD 程序
DA VA[750],VB[750]	定义数组
Count=0	初始化计数器
n=0	初始化位置增量
# LOOP	循环标号
VA[count]=n	填数组 VA
VB[count]=n	填数组 VB
n=n+10	增量位置
count= count+1	计数器+1
JP# LOOP, Count<750	如果数组未满, 则循环
#A	标号
LM AB	指定 AB 轴为直线插补方式
count=0	初始化数组计数器
# LOOP2; JP# LOOP2; -LM=0	若程序缓冲区满, 则等待
JS# C, Count=500	在第 500 个插补线段处开始运动
LI VA[count], VB [count]	指定直线插补线段
count= count+1	数组计数器+1
JP # LOOP2, Count<750	重复循环, 直到数组执行完
LE	直线插补结束
AMS	插补程序执行完后
MG"DONE"	发送信息
EN	结束程序
# C; BG; EN	开始执行运动子程序

5. 直线和圆弧插补方式

DMC-2×00 控制器提供了现成的 2 维圆弧插补功能, 直线插补和圆弧插补可以混合编程。即使在直线和圆弧之间的过渡处, 沿轨迹的运动也是以矢量插补速度连续进行。DMC-2×00 执行所有直线、圆弧插补复杂运算, 上位 PC 机可脱机去处理其他任务。而对于 3 维以上的圆弧插补, 可以用高级语言 (如: C/C++、VC 等), 并调用 API 函数, 在开放式平台上进行第二次开发, 以满足这方面的应用要求。

联动运动方式类似于直线插补方式。可以选择任意两个轴为联动轴构成直线和圆弧插补运动。此外, 也能对第三轴进行控制, 使其与选择的两轴运动保持正切运动关系。**注意:** 在任何时候, 只能为联动轴指定任意两个轴构成两维圆弧插补关系。

命令 VM m、n、p, 其中 m 和 n 是两个联动轴, p 是正切轴。

注意: 不需要用逗号 (,) 分开 m, n 和 p, 例如, VM ABC 选择 AB 轴为联动轴, 而 C 轴为正切轴。

5.1 指定联动坐标平面

DMC-2×00 为直线插补和圆弧插补方式提供了两组不同的坐标系。他们分别用字母 S 和 T 加以区别。

要指定矢量命令, 必须先区分坐标平面。用命令 CAS 来分辨 S 平面, 用命令 CAT 来分辨 T

平面。所有矢量命令都适用于当前有效的坐标系，直到用 CA 命令加以改变。

5. 2 指定矢量线段

运动线段用两个命令进行描述，它们分别是：VP 为直线插补线段，CR 为圆弧插补线段。一旦规定了一组直线插补线段和（或）圆弧插补线段，就要用命令 VE 来结束。这就意味着这些命令所构成的程序为联动轴运动。在执行第一个联动运动之前，控制器先将联动程序中所有运动轴的当前位置定义为 0。

注意：局部 0 定义不影响绝对坐标系或以后的联动运动程序。

VP xy 命令规定矢量运动终点相对于起点的坐标值，没有后续轴就不需要用逗号来终止。命令 CRR, q, d 以半径 r，起始角 q 和弧角 d 来定义圆弧的弧长。起始角 q 约定为：0，对应于轴正方向，而对于 q 和 d，逆时针方向旋转为正向。

在一个程序中，可指定 511 个 CR 或 VP 插补线段，而且必须用命令 VE 来结束；用 BGS 来启动矢量运动程序，一旦开始运动，就可以加入附加插补线段。

在运动开始之前能用 CS 命令来删除存储在缓冲区中的 VP 和 CR 命令。用指令 STS 或 AB1 来停止运动。STS 命令以规定的减速度使运动停止；AB1 立即使运动中止。

必须用矢量结束命令来指定联动运动结束，此命令需要控制器减速停止在最后一插补运动之后。若未给定 VE 命令，就必须用 AB1 来中止联动控制程序执行。

用户有必要在 DMC-2×00 程序缓冲区中存入足够的运动程序段，以确保连续运动。如果控制器未接收到附加运动程序段，且没有 VE 命令，控制器就会立即使运动停止在最后的矢量处，也不会有可控制的减速度过程。LM ? 或 -LM 读取能够发送到缓冲区中的运动线段的可用空间。返回值为 511 表示缓冲区是空的，此时可发送 511 个插补线段；0 表示缓冲区是满的，此时不能发送附加插补线段。只要缓冲区未满，就能以 PC 机总线速度发送附加插补线段。

用操作数-CS 来确认插补线段计数器的值。

5. 3 附加命令

命令 VS_n, VA_n 和 VD_n 用于设置矢量速度，加速度和减速度。

命令 VT 是用于联动运动的 S 曲线平滑参数。

5. 3. 1 对各插补线段指定矢量速度

可以用即时命令 VS 指定矢量速度。也能将它用如下指令附加到插补线段中：

VP a,b <n>m

CR r, θ , δ <n>m

第一个参数 <n>，等效于在给定插补线段起点加入命令 VS_n，它会使运动向新命令的速度加速，若无其他限制。

第二个参数 >m 规定了到达插补线段终点处的矢量速度值。

注意：参数 >m 可以使运动在给定的插补线段内或在前一插补线段期间开始减速，在给定 VA 和 VD 值条件下，满足最终的速度要求。

但是，请注意，控制器每次只能以一个 >m 命令工作，因此，一个参数可以用另一个进行屏蔽；例如，如果参数 >100000 由 >5000 跟随其后，且减速距离不够，那么就不满足第二个条件，此时，控制器就设法将速度减到 5000，但会到达不同点。

5. 3. 2 改变进给率（进给率修调）

命令 VR_n 使进给率 VS 以 0.0001 分辨率在 0~10 之间进行比例修调。此命令立刻起作用并使 VS 成比例修正。VR 也适用于用“<”操作数指定矢量速度的场合。此功能对于像 CNC 那样要求有进给率修调功能的应用非常有用。VR 不对加，减速度做比例处理。例如，VR 0.5 使 VS2000 除以 2。

5. 3. 3 以编码器分辨率补偿偏差

当使用矢量方式时，DMC-2×00 对编码器分辨率采用缺省比例系数 1：1，如果实际情况与此不同，就用命令 ES 对编码器计数进行比例处理；ES 命令附带两个参数，分别代表用于矢量运动的两个编码器的计数值。两个数的比例越小就会由更高的分辨率编码器来倍乘。通常称此为椭圆缩放系数，此命令对于椭圆加工非常有用。详见第二篇命令手册中 ES 命令。

5. 3. 4 条件启动

AVn 命令是在矢量运动完成后的条件启动，在执行程序中的下一条命令之前，等待相对矢量距离 n 到达。

5. 3. 5 正切运动

像玻璃切割，石材切割类应用，要求第三轴（即刀片）与切割平面运动轨迹保持正切，为了满足这类应用，DMC-2×00 控制器使一个轴做为正切轴，用 VM 命令后面的参数来定义联动轴和正切轴。

VM m, n, p 其中，m, n 指定联动坐标轴，p 指定正切轴，如 A, B, C 或 D
P=N 使正切轴关断。

在正切方式工作之前，需要用 VM 命令来分配一个轴，并通过 TN m, n 命令定义该轴的偏置和比例系数；m 为比例系数，单位为 cts/度，n 为联动坐标平面中等于 0 度的正切位置。用操作数 -TN 来返回正切轴的初始位置。

5. 4 联动控制程序命令汇总

命 令	说 明
VM m, n, p	指定平面二维联动轴，其中 m 和 n 代表平面轴，p 是正切轴
VP m, n	矢量终点坐标值，其中：m=A,B,C,D
CR r θ δ	指定圆弧插补线段，其中：r 为半径，θ 为起始角，δ 为弧角，逆时针为正
VS n	指定矢量速度或进给率
VA n	指定矢量加速度
VD n	指定矢量减速度
VR n	指定矢量速度比例系数
BGS	开始矢量运动
CS	清除缓冲区中的插补程序
AV n	在相对矢量距离 n 到达之后的条件启动
AMS	等待直到运动程序完成后才再执行下一条命令
TN m, n	正切缩放和偏置
ES m, n	椭圆比例系数
VT	联动轴 S 曲线平滑常数
LM ?	读取控制器程序缓冲区中直线和圆弧插补线段可用空间，0 代表满，512 代表零

5. 5 联动运动程序操作数汇总

操 作 数	说 明
-VPM	读取最后交点处的绝对坐标值
-AV	已运动的矢量距离
-LM	读取控制器程序缓冲区中直线和圆弧插补线段可用空间，

	0 代表缓冲区满，512 代表缓冲区空
-CS	插补线段计数器——程序中插补线段数，以 0 开始计数
-VE	联动坐标运动程序的矢量长度

当 AV 用作为操作数时，-AV 返回矢量程序所运动的距离。

能够用操作数-VPA 和-VPB 来读回沿轨迹所规定的最后一点的坐标值。

5. 6 举例

举例1 正切轴

假设一个 X Y 工作台，带有控制切刀的 C 轴，C 轴编码器为 2000cts/rev，在上电之后它对 C 轴初始化，使切刀指向+B 方向，要进行 180° 圆弧切割，半径为 3000，中心点在坐标原点，起始点在 (3000, 0)，以逆时针方向运动，终点在 (-3000, 0)。注意：XY 平面的 0° 位置是在+A 轴方向，对应于 Z 轴位置-500，定义偏置。此运动分两部分，首先，将 A,B,C 轴驱动到起点，然后，使切刀运动。假使切刀用输出位 0 来使能。

指 令	说 明
# EXAMPLE	程序标号
VM ABC	具有 C 轴为正切轴的 A B 轴联动控制
TN 2000/360, -500	在 A B 平面，2000cts/度，位置-500 为 0°。
CR 3000, 0, \$0	半径为 3000，起始角为 0°，逆时针转 180°
VE	矢量结束
CB 0	切刀使能断开
PA 3000, 0, -TN	把 A B 轴移到起点位置，C 轴提到初始正切位置。
BG ABC	开始运动
AM ABC	当运动完成时，
SB 0	使切刀使能
WT 50	使切刀使能等待 50msec
BG S	开始圆弧切割
AM S	当联动轴运动完成时
CB 0	使切刀使能关断
MG"ALL DONE"	
EN	程序结束

举例2 坐标联动运动

运动轨迹如图6.3所示，进给率为20000 cts/sec，运动平面为AB。

指 令	说 明
VM AB	指定运动平面
VS 20000	指定矢量速度
VA 100000	指定矢量加速度
VD 100000	指定矢量减速度
VP -4000,0	线段AB
CR 1500,270,-180	线段BC
VP 0,3000	线段CD
CR 1500,90,-180	线段DA
VE	矢量程序结束
BGS	开始运动

此运动以A点开始，向点B, C, D, A运动，假设，运动到中途A B.两点之间时，查询控制器。

_AV的值是2000

_CS的值是0

_VPA和 _VPB是A点的绝对坐标值

假设在C 、D.两点中间再次查询：

_AV的值是 $4000+1500\pi+2000=10,712$

_CS 的值是2

_VPA和_VPB为C点的坐标值。

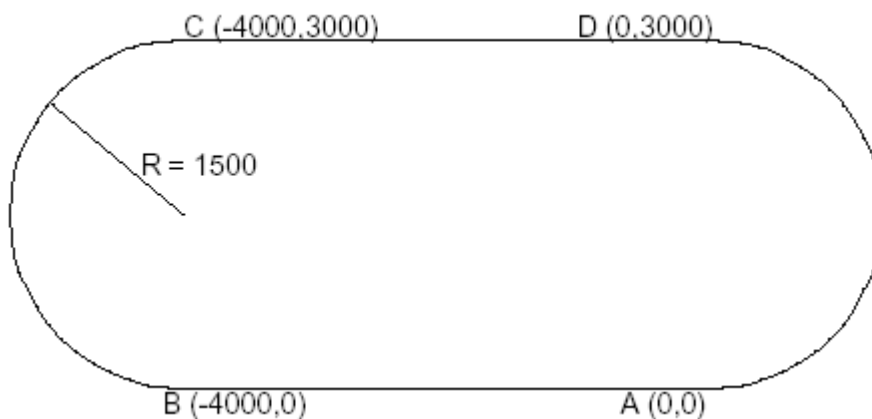


图6.3 – 运动轨迹图

6. 电子齿轮同步功能

此方式具有8个轴从电气上与另外一些主动轴实现同步控制功能。主动轴可以两个方向旋转，而其它电子齿轮轴则以规定的齿轮比跟随运动。各轴的齿轮比可以不同，在运动期间可以改变。

命令GA ABCDEFGH指定主动轴，GR a,b,c,d指定从动轴的齿轮比，其中，齿轮比可以是分辨率为0.0001~±127.9999之间的数。电子齿轮有两种方式：分别是标准电子齿轮同步和龙门同步方式。用命令GM使龙门同步方式有效。GR 0,0,0,0关掉两种方式电子齿轮同步控制。在标准方式中，限位开关或ST命令会使电子齿轮同步功能无效，但在龙门同步方式中，却不起作用。

命令GM a,b,c,d选择在龙门同步方式下要控制的轴。参数1使龙门同步方式使能，而0 则使其不起作用。

GR使指定的轴与主动轴的实际位置成电子齿轮耦合关系，主动轴用运动命令PR, PA 或JG等来命令。.

当控制器以JOG方式或独立轴运动方式驱动主动轴时，就可以将主动轴定义为该轴的命令位置，而不是实际位置。用字母C来表明主动轴是命令位置。例如，GACA表示电子齿轮轴是以A轴的命令位置来同步。

另一种电子齿轮方法，是从动轴电机与GAS所规定的几个轴的命令矢量位置相同步。例如，如果A轴和B轴形成圆弧运动，C轴可以与矢量运动成比例运动。同样，如果A、B、C轴执行直线插补运动，就能使W轴与矢量运动成电子齿轮运动关系。

电子齿轮功能也允许电子齿轮轴电机执行除电子齿轮之外的第二个独立轴运动或联动运动。例如，当电子齿轮轴电机以1:1的比率跟随主动轴时，也可以用PR、JG命令或VP、LI命令使其超前一个附加的距离。

6. 1 电子齿轮同步功能命令汇总

命 令	说 明
GA n	为电子齿轮定义主动轴，其中：
	n = A,B,C,D,E,F,G,H主编码器为主动轴
	n = CA, CB, CC, CD, CE, CF,CG,CH为命令位置
	n = DA, DB, DC, DD, DE, DF,DG,DH为辅助编码器
	n = S或T为联动运动
GR a,b,c,d,e,f,g,h	为从动轴设置齿轮比，0 使所指定轴的电子齿轮无效
GM a,b,c,d,e,f,g,h	1设置为龙门同步方式，0 使龙门同步无效
MR a,b,c,d	经过指定值时，使运动反向的条件启动，只能使用1个域
MF a,b,c,d	经过指定值时，使运动正向的条件启动，只能使用1个域

6. 2 举例

举例1 简单主从运动

主动轴以100000cts/sec速度移动10000cts，定义B轴为主动轴，A、C、D轴与主动轴成电子齿轮运动，齿轮比分别为5,-0.5，10。

指 令	说 明
GA B,,B,B	指定主动轴为B轴
GR 5,,-0.5,10	设置齿轮比
PR ,10000	指定B轴位移量
SP ,100000	指定B轴速度
BGB	开始运动

举例2 电子齿轮

目的：以外部主动轴速度的1.132倍和-0.045倍运行两个电子齿轮电机，主动轴以0~1800r/min (2000 cts/rev编码器)速度驱动主动轴。

方案：使用DMC-2x30控制器，其中C轴为主动轴，A、B轴为电子齿轮轴。

指 令	说 明
MO C	关断C轴使能，作为主动轴使用
GA C, C	指定C轴为A、B轴的主动轴
GR 1.132,-.045	指定齿轮比

现在假设：A轴的齿轮比要随时改变为2，用以下命令来实现：

GR 2	指定A轴齿轮比为2
------	-----------

举例3 龙门同步方式

在主动轴和随动轴由DMC-2x00控制器的应用中，想使随动轴与主动轴的命令位置同步，而不是实际位置。忽略可能引起振荡的各轴之间的联接。

例如，由A、B两轴在两侧驱动龙门架，对于两台电机之间的刚性连接而言，就需要龙门同步方式。A轴是主动轴，B轴是随动轴，要使B轴与A轴的命令位置同步，使用命令：

指 令	说 明
GA CA	指定A轴命令位置为B轴的主动轴
GR 1	设定B轴的齿轮比为1:1
GM 1	设置龙门同步方式
PR 3000	指定A轴位置
BG A	启动A轴运动

在电子齿轮方式中，您也可以进行轮廓位置修正。例如，假设，您需要使从动轴超前10cts，就简单地命令：

IP ,10	指定B轴增量位置位移量10
--------	---------------

在这些条件下，此IP命令等效于：

PR,10	指定B轴相对位移量10
BGB	开始B轴运动

修正量往往很大，这些需要通常发生在使切刀或输送带同步控制时。

举例4 使具有轮廓速度修正功能的两个输送带同步

指 令	说 明
GA,A	定义A轴为B轴的主动轴
GR,2	设置B轴齿轮比为2:1
PR,300	指定修正距离
SP,5000	指定修正速度
AC,100000	指定修正加速度
DC,100000	指定修正减速度
BGB	开始修正

7. 电子凸轮

电子凸轮是一种使几个运动轴能够周期性同步的运动控制方式。7个轴可以作为一个主动轴的从动轴。主动轴编码器必须通过主编码器输入。

电子凸轮是更为典型的电子齿轮应用，它以轴之间的凸轮表关系为基础，使所有控制轴实现同步。例如，DMC-2x80控制器就可以有1个主动轴和7个从动轴。为了说明创建凸轮表，我们考虑A轴为主动轴，B轴为从动轴的凸轮关系。其图形关系见图6.4。

第1步. 选择主动轴

电子凸轮方式的第一步是选择主动轴，用如下命令来进行：

EAp 其中：p = A、B、C、D
P是选择的主动轴

为了举例方便，由于主动轴为X轴，因此，我们规定EAA。

第2步. 规定主动轴循环周期和从动轴变化

在电子凸轮方式中，主动轴的位置总是表达为1个循环周期模。在本例中，X轴位置总是表达为0~6000之间的数值。同样，也对从动轴重新定义，以便它在0处开始启动，在1500处结束。

在循环终点，当主动轴为6000，从动轴为1500时，就将a和b的位置重新定义为0，为了指定主动轴循环周期和从动轴循环周期变化，我们使用指令EM。

EM a, b, c, d

其中：a, b, c, d 指定主动轴的循环周期和一个循环周期后从动轴的总变化量。

主动轴的循环周期限定在8388607，而每个循环周期从动轴变化量限定在 2147483647。如果变化量是负数，就取其绝对值。对于给出的例子，主动轴的循环周期为 6000 cts，而从动轴的变化量为1500。因此，我们使用命令：

EM 6000,1500

第3步. 指定主动轴间隔和起始点

下一步，我们要创建电子凸轮表，以主动轴位置的均匀间隔规定凸轮表。最多可以有256个间隔。用如下命令来规定主动轴间隔大小和起始点：

EP m,n

其中：m是间隔宽度，单位为cts，n是起始点。

对于给定的例子，我们通过规定主动轴间隔点 0, 2000, 4000, 6000来构建凸轮表。因此，规定如下：

EP 2000,0

第4步. 指定从动轴位置

下一步，我们用如下指定规定从动轴位置：

$$ET[n] = a, b, c, d$$

其中：n表示各点次序

n值以0开始，最多到256，参数a，b，c，d表示相应点从动轴位置。对于此例而言，表达如下：

$$ET[0] = 0$$

$$ET[1] = 3000$$

$$ET[2] = 2250$$

$$ET[3] = 1500$$

这样就创建ECAM表。

第5步. 使ECAM使能

为了使ECAM方式使能，使用命令

$$EB\ n$$

其中：n=1 使ECAM方式使能，n=0使ECAM方式无效。

第6步. 使从动轴运动啮合

要使从动轴啮合，使用指令

$$EG\ a, b, c, d$$

其中：a，b，c，d是必须使相对应的从动轴啮合处的主动轴位置。

如果任意参数值在一个循环周期范围之外，凸轮就立即啮合。当凸轮啮合时，重新定义从动轴位置，模为1个循环周期。

第7步. 使从动轴运动脱开

要使凸轮脱开，使用如下命令：

$$EQ\ a, b, c, d$$

其中：a，b，c，d是脱开相对应从动轴处的主动轴位置。

这就使从动轴在指定的主动轴位置点脱开。如果参数在主动轴循环周期以外，就可瞬时停止。

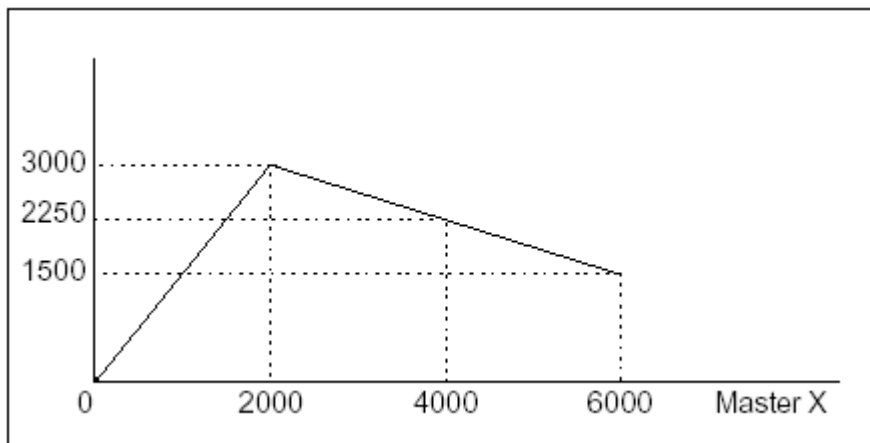


图6.4-电子凸轮举例

第8步. 创建产生ECAM表的程序

为了表明整个过程，用下式来表示凸轮关系：

$$B = 0.5 * A + 100 \sin(0.18 * A)$$

其中：A是主动轴，循环周期为2000cts

凸轮表能手动一点一点创建，或用程序自动创建。下面的程序包含配置，指定EAA将A轴定义为主动轴。主动轴的循环周期是2000。循环一到，A轴就变化1000。因此，命令为：

EM 2000,1000.

假设，我们用100个线段定义一个表，因此，增量间隔就为20cts。

如果主动轴在0点开始起动，那么，所需要的指令就是：EP 20,0.

下面的程序计算表点，由于相位角等于0.18A，且A以20增量变化，因此，相位变化增量为3.6°，然后，程序就按照方程式计算B的值，并将所得值用指令ET[N] = ,B分配给表。

指 令	说 明
# SETUP	标号
EAA	选择A为主动轴
EM 2000,1000	凸轮周期
EP 20,0	主动轴位置增量
N = 0	标志
# LOOP	循环从方程式建立凸轮表
P = N*3.6	注：3.6 = 0.18*20
S = @SIN [P] *100	定义正弦位置
B = N *10+S	定义从动轴位置
ET [N] = , B	定义凸轮表
N = N+1	计数器加1
JP # LOOP, N<=100	重复循环
EN	程序结束

第9步. 创建程序运行ECAM方式

现在假设，从动轴用启动信号输入点1 啮合，但啮合和脱开点必须在循环周期终点A= 1000，B = 500处进行，这样就必须驱动B轴到该点以避免跳动。

用程序实现如下：

指 令	说 明
# RUN	标号
EB1	使凸轮使能
PA,500	起始位置
SP,5000	B轴速度
BGB	使B轴电机运动
AM	在B轴运动之后
AI1	等待起动信号
EG,1000	使从动轴啮合
AI - 1	等待停止信号
EQ,1000	使从动轴脱开
EN	结束

7. 1 ECAM命令汇总

命 令	说 明
EA p	指定ECAM主动轴，其中p =A, B, C或D。p为所选主动轴
EB n	使ECAM使能
EG a, b, c, d	使ECAM啮合
EM a, b, c, d	指定主动轴循环周期和从动轴一个循环周期中的变化量
EP m,n	定义ECAM表入口大小和编程
EQ m,n	使ECAM在指定位位置脱开

ET[n]	定义ECAM表入口
-------	-----------

7. 2 ECAM操作数汇总

命 令	说 明
_EB	读取 ECAM 状态
_EC	读取当前 ECAM 标志
_EGa	读取各轴 ECAM 状态
_EM	读取各轴 ECAM 循环周期
_EP	读取 ECAM 表间隔值
_EQx	读取各轴 ECAM 状态

7. 3 举例

举例1 ECAM

下例表示具有主动轴为C轴，两个从动轴为A、B轴的ECAM 程序：

指 令	说 明
# A; V1=0	标号；初始化变量
PA 0,0; BG AB; AM AB	A、B轴运动到位置0, 0
EA C	C轴作为ECAM主动轴
EM 0,0,4000	C轴变化量为4000，A、B轴为0
EP400,0	ECAM间隔为400 cts，以0为始点
ET[0]=0,0	主动轴在0位置时，为第1点
ET[1]=40,20	ECAM表中的第2点
ET[2]=120,60	ECAM表中的第3点
ET[3]=240,120	ECAM表中的第4点
ET[4]=280,140	ECAM表中的第5点
ET[5]=280,140	ECAM表中的第6点
ET[6]=280,140	ECAM表中的第7点
ET[7]=240,120	ECAM表中的第8点
ET[8]=120,60	ECAM表中的第9点
ET[9]=40,20	ECAM表中的第10点
ET[10]=0,0	下一个循环周期的起始点
EB 1	ECAM方式使能
JGC=4000	设置C轴为JOG方式，速度为4000cts/sec
EG 0,0	当主动轴= 0时，使A、B轴啮合
BGC	开始C轴JOG运动
# LOOP; JP# LOOP,V1=0	一直循环到设置变量
EQ2000,2000	当主动轴= 2000时，使A、B轴脱开
MF,, 2000	等待主动轴到达2000
ST C	停止C轴运动
EB 0	退出ECAM方式
EN	程序结束

上例表示了创建ECAM程序的方法及将命令发送给控制器的方法。图6. 5是用WSDK工具软件捕获到的各轴运动曲线，图6. 5中的第一个曲线是A轴，第二个曲线是B轴，第三个是C轴。

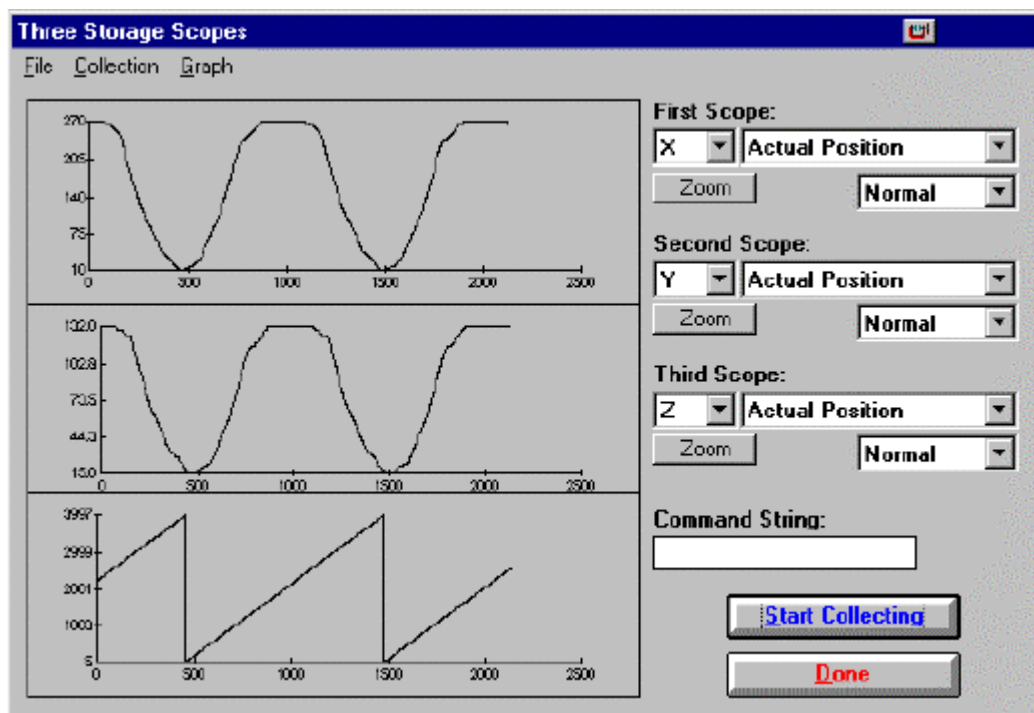


图6.5 – A B C速度轮廓

8. 轮廓方式

DMC-2x00也具有轮廓功能，用此功能，用户能够在1~8轴内定义任意位置曲线，从而非常理想地跟踪由计算机产生的轨迹，如：抛物线、螺旋线或用户自定义的轮廓。此功能对轨迹类别无限制，可以是直线或圆弧线段，轨迹长度也无限制。

8. 1 指定轮廓线段

用命令CM来指定轮廓方式，例如，CM AC将A、C轴指定为轮廓加工。没有用于轮廓方式的其它轴可以以其它方式工作。

轮廓用位置增量加以描述，用命令CDa, b, c, d.及与之相对于时间间隔 DT n来定义位置增量。参数n指定时间间隔。时间间隔定义为 2^n ms，其中：n为1~8。控制器在指定的增量之间进行直线插补，对增量的插补按各毫秒产生一个点来进行。

例如，我们以图6.6为例，由以下各点来描述A轴位置。

- | | |
|----|---------------|
| 点1 | T=0ms时，A=0 |
| 点2 | T=4ms时，A=48 |
| 点3 | T=12ms时，A=288 |
| 点4 | T=28ms时，A=336 |

同样的轨迹用增量表达如下：

- | | | | |
|-----|--------|-------|------|
| 增量1 | DA=48 | 时间=4 | DT=2 |
| 增量2 | DA=240 | 时间=8 | DT=3 |
| 增量3 | DA=48 | 时间=16 | DT=4 |

当控制器收到沿这些点产生轨迹的命令时，它就在这些点之间进行直线插补。插补各点的结果为：在1ms处,位置值12，在2ms处，位置值24，等等，以此类推。关于上例中的程序如下：

指	令	说	明
#A		标号	

CM A	指定A轴为轮廓方式
DT 2	指定第1个时间间隔为 2^2 ms
CD 48; WC	指定第1个位置增量
DT 3	指定第2个时间间隔为 2^3 ms
CD 240; WC	指定第2个位置增量
DT 4	指定第3个时间间隔为 2^4 ms
CD 48; WC	指定第3个位置增量
DT0; CD0	退出轮廓方式
EN	程序结束

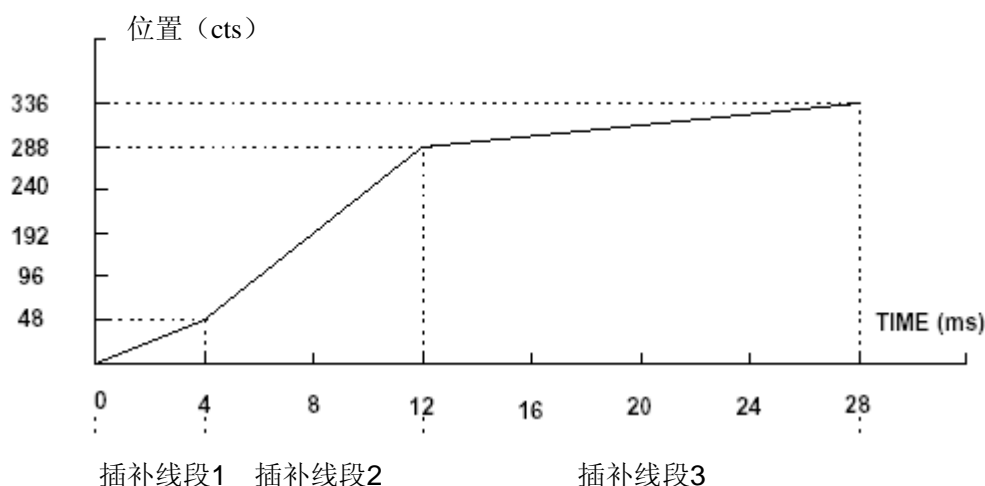


图 6.6 – 单轴轮廓方式轨迹图

8. 2 附加命令

命令WC用作轮廓方式时的条件启动，它意味着等待时间到达结束前一命令执行时再执行后续命令。用DT0和CD0配合使用就退出轮廓方式。

如果控制器未发现新的数据记录，且仍在轮廓方式，控制器就等待新数据。在等待当中，控制器不产生新的运动命令。如果控制器接收到错误数据，就以问号（？）回应。

8. 3 轮廓方式命令汇总

命 令	说 明
CM ABCDEFGH	指定用于轮廓方式的坐标轴，任何非轮廓轴可以工作在其它方式
CD a, b, c, d, e, f, g, h	按时间间隔指定位置增量，增量范围为 ± 32000 。在CD0与DT0配合使用时，就使轮廓方式结束。
DTn	对位置增量指定时间间隔，其中n是1~8之间的整数。0使轮廓方式结束，如果n不改变，就不需要与各条CD命令一同指定。
WC	在处理下一个数据记录之前，等待前一时间间隔完成

8. 4 通用速度轮廓

轮廓方式非常适用于产生任意速度轮廓。可以把速度轮廓指定为数学函数或点的汇集。设计包含两部分：产生用于数据点的数组和运行程序。

8. 5 举例

举例1 产生数组

我们先看看图6.7中所示的速度和位置轮廓曲线。目的是在120ms内，使电机运动6000 cts，速度轮廓为正弦波曲线，以减少冲击和系统振荡。如果我们以Bms内的A cts描述位移量，我们就能用下式描述运动方程。

$$\omega = \frac{A}{B} (1 - \cos(2\pi T / B))$$

$$X = \frac{AT}{B} - \frac{A}{2\pi} \sin(2\pi T / B)$$

注: ω 是角速度; A是位置; T是变量。时间以ms为单位。

在此例中, A=6000, B=120, 位置、速度轮廓方式为:

$$A = 50T - (6000/2\pi) \sin(2\pi T/120)$$

注意: 速度 ω 以cts/ms为单位, 方程式如下:

$$\omega = 50 [1 - \cos 2\pi T/120]$$

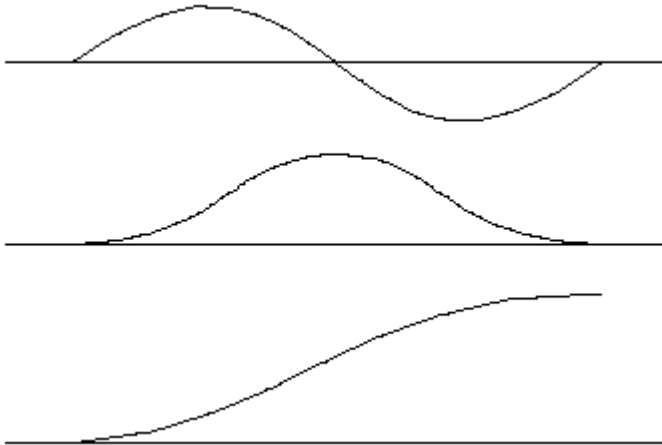


图6.7 – 具有正弦波加速的速度轮廓

DMC-2x00具有三角函数运算功能, 不过, 必须以度表达参数。结合我们的举例, A的方程式表达为:

$$A = 50T - 955 \sin 3T$$

以下是本例中产生轮廓运动的完整程序。为产生数组, 我们以8ms的间隔计算位置值, 并将其存在数组POS, 然后, 计算各位置之间的差值, 存入数组DIF, 最后, 以轮廓方式运转电机。

轮廓方式举例程序

指 令	说 明
# POINTS	标号, 定义A轴位置点的程序
DM POS[16]	分配存储器
DM DIF[15]	
C=0	设置初始条件, C是标志
T=0	T是时间, 以ms为单位
# A	
V1=50*T	
V2=3*T	以度为单位的参数
V3=-955*@SIN[V2]+V1	计算位置
V4=@INT[V3]	V3值取整

POS[C]=V4	存入数组POS
T=T+8	
C=C+1	
JP #A,C<16	
#B	求位置差的程序标号
C=0	
#C	
D=C+1	
DIF[C]=POS[D]-POS[C]	计算差值并存储
C=C+1	
JP #C,C<15	
EN	第一个程序结束
#RUN	运转电机的程序标号
CMX	轮廓方式
DT3	4 ms间隔
C=0	
#E	
CD DIF[C]	轮廓距离存入DIF
WC	等待完成
C=C+1	
JP #E,C<15	
DT0	
CD0	停止轮廓方式
EN	程序结束

举例2 示教（录返功能）

有些应用需要对机器运动轨迹进行示教，对此，用DMC-2x00数组自动捕获功能来捕获位置数据即可实现示教，然后以轮廓方式对捕获的数据进行录返。使用如下数组命令：

指 令	说 明
DM C[n]	规划数组
RA C[]	为自动记录规定数组(DMC-2x40为最多4个)
RD _TPA	指定捕获的数据(如 _TPA 或 _TPC)
RC n,m	指定捕获时间间隔，其中n为2 ⁿ 采样时间，m是捕获到的记录数值
RC? 或 _RC	如果在记录，返回值为1

录返举例程序：

指 令	说 明
#RECORD	程序标号
DP0	将A轴位置定义为0
DA*[]	清空所有数组
DM xpos [501]	分配调用xpos 的501个元素数组
RA xpos []	将元素记录入xpos 数组
RD _TPA	要记录的元素为A轴编码器位置
MOA	A轴使能关断
RC2	以2 ² ms采样周期开始记录

# LOOP1; JP# LOOP1,_RC1	循环直到所有元素都被记录
# COMPUTE	确定连续点之间差值的子程序
DM dx [500]	为保持轮廓点分配500个元素数组
i = 0	设置循环计数器
# LOOP2	循环计算差值
dx[i]= xpos [i +1]- xpos [i]	计算差值
i = i +1	循环计数器加1
JP# LOOP2, i <500	继续循环，直到dx置满
# PLAYBK	录返运动子程序
SHX	伺服使能
WT1000	等待1 sec (1000 msec)
CMA	指定A轴为轮廓方式
DT2	设置轮廓数据采样间隔为22 msec
i =0	设置数组标志为0
#LOOP3	执行轮廓点的子程序
CD dx[i]; WC	轮廓数据命令；等待下一个轮廓点
i = i +1	标志加1
JP# LOOP3, i <500	循环，直到执行完所有数组元素
DT0	设定轮廓更新率为0
CD0	轮廓方式关断 (与 DT0配合使用)
EN	程序结束

关于自动数组捕获的其它内容，请参见第7章中“数组”一节

9. 虚拟轴功能

DMC-2x00 控制器具有虚拟轴功能，指定为 N 轴。此轴没有编码器也没有 DAC 输出。不过，能够用如下命令对其编程：

AC、DC、JG、SP、PR、PA、BG、IT、GA、VM、VP、CR、ST、DP、RP、EA 虚拟轴的主要用途是在 ECAM 方式中设置虚拟主动轴，以及执行矢量方式的不需要的部分运动。这些例子通过下面一些例子加以说明。

举例1 ECAM 主动轴举例

假设：强制 A、B 轴沿着由 ECAM 表所描述的轨迹运动，并假定，ECAM 主动轴不是外部编码器但必须是可控制的变量。

通过用命令 EAN 把 N 轴定义为主动轴，并用命令设置主动轴的模（例如 EMN=4000）来实现这一要求，其次，创建凸轮表，要使强制轴运动，就简单地以 JOG 方式或命令 PR、PA 命令 N 轴。

例如：PAN=2000

BGN

会使 A、B 轴按运动周期运动到相对应的点。

举例2 正弦波运动举例

假设：X 轴必须进行 10 个循环的正弦波运动，幅值为 1000cts，频率为 20Hz，据此要求，我们让 A 轴和 N 轴进行圆弧运动，注意：VS 的值必须是：

$$VS=2\pi \cdot R \cdot F$$

其中 R 是半径或幅值，F 是频率，单位为 Hz。

设置 VA 和 VD 为最大值，以求最快的加、减速度。

指 令	说 明
VMAN	选择轴
VA68000000	最大加速度
VD68000000	最大减速度
VS125664	VS 为 20Hz
CR1000, -90, 3600	10 圈
VE	
BGS	开始矢量运动

10. 步进电机工作方式

将控制器配置为步进电机工作方式时，有几个命令与伺服方式有不同的解释。下面我们就对步进电机工作方式的不同点加以说明。

10.1 设置步进电机工作方式

为了使控制器工作在步进电机方式，在硬件方面，必须插入用于步进电机方式设定的短路棒，详见第 2 章。

在硬件设置完成之后，用命令 MT 从软件功能上进行设置。用于命令 MT 的参数如下：

- 2 设置步进电机为有效低输出脉冲
- 2 设置步进电机为有效高输出脉冲
- 2.5 设置步进电机为有效低输出脉冲和方向输出
- 2.5 设置步进电机为有效高输出脉冲和方向输出

10.2 步进电机平滑处理

命令 KS 具有步进电机平滑处理功能。平滑处理的效果可以视为一个简单的阻-容（单极性）滤波器。滤波器位于运动轮廓分配器之后，使脉冲输出间隔发生变化，从而使步进电机工作更为平稳，因此起到平滑作用。当步进电机以满步或半步工作时，KS 的作用最为明显。KS 会使步进脉冲按照所规定的时间常数进行延迟。当以步进电机方式工作时，总是要用 KS 设置一定量的平滑处理；由于滤波效果出现在轮廓分配器之后，因此，在所有步进脉冲通过滤波器之前，轮廓分配器就在准备其它后续运动。如果在前一运动完成之前就命令控制器产生其它后续运动，由于可能丢失脉冲，因此，加入平滑处理是非常重要的。参见下一节的讨论。

也能够使用一般的运动平滑处理命令 IT，命令 IT 的用途是使运动轮廓输出更为平滑，减小由于加速度而引起的冲击。

10.3 监视产生脉冲——命令脉冲

为了控制器工作正常，就需要确保在进行其它后续运动之前，控制器已完成产生所有步进脉冲。如果步进电机来回运动，这一点就尤为重要。例如，当以伺服电机方式工作时，就用条件启动命令 AM 来确认运动轮廓分配器何时完成并何时准备执行新的运动命令。不过，在步进电机方式工作下，当运动轮廓分配完成时，控制器可能还在产生步进脉冲；这是由于步进电机平滑滤波器而引起。为了更好地理解这点，我们看看控制器进行产生步进脉冲的过程。

首先，控制器按照运动命令产生运动轮廓。

第二，轮廓分配器产生由运动轮廓所指令的脉冲。由运动轮廓分配器产生的脉冲可以用命令 RP（参考位置）来监控，RP 读取的是由运动轮廓分配器所确定的位置绝对值。用 DP 命令来设置参考位置值；例如，DP0，将 A 轴的参考位置定义为 0。

第三，运动轮廓分配器的输出由步进平滑滤波器进行滤波。此滤波器将延迟加入步进电机脉冲的输出中。延迟量取决于用命令 KS 所规定的参数。如前所述，总会有一些步进电机平滑量。KS 的缺省值是 2，此值对应于 6 个采样周期的时间常数。

第四，对步进电机平滑滤波器的输出进行缓冲，然后输出到步进电机驱动器。能用命令 TD(Tell Dual) 监视由平滑滤波器产生的脉冲。TD 返回由缓冲实际输出所决定的位置绝对值，DP 命令设置步进计数寄存器的值以及参考位置值。例如，DP0 将 A 轴的参考位置设置为 0。

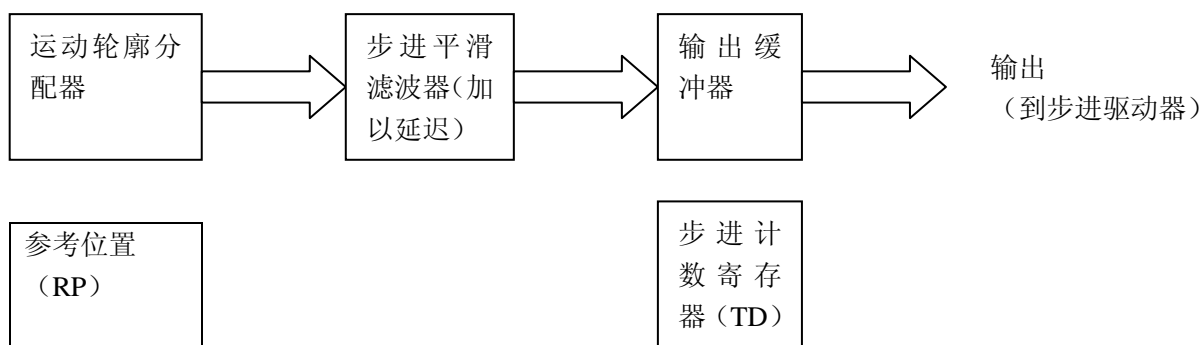


图 6.8

10. 4 运动完成条件启动

在步进电机方式中，用命令 MC 使执行处理暂停，只有当命令位置中所指定的脉冲数与步进寄存器的输出脉冲数相同时，控制器才会继续执行后续命令。一般来说，MC 条件启动命令比 AM 条件启动命令更为有用，原因在于：由于步进电机平滑功能，步进脉冲从命令位置开始能够得到延迟。

10. 5 编码器用于步进电机

可以将编码器用于步进电机上，来检查电机实际位置。如果使用编码器，就必须将其连接到主编码器输入口。

注意：当以步进电机方式工作时，不能使用辅助编码口。编码器的位置可以用命令 TP 来查询；用命令 DE 能定义位置值。

注意：步进电机不能以闭环方式工作。

10. 6 步进电机工作方式命令汇总

命 令	说 明
DE	定义编码器位置（当使用编码器时）
DP	定义参考位置和步进计数寄存器
IT	运动轮廓平滑时间常数——独立运动时间常数
KS	步进电机平滑常数
MT	指定电机类型（用于步进电机时，为 2，-2，2.5，-2.5）
RP	读取命令的位置
TD	读取由控制器产生的步进脉冲数
TP	读取编码器位置

10. 7 步进电机工作方式操作数汇总

操 作 数	说 明
_DEa	读取 ‘a’ 轴步进计数寄存器的值
_DPa	读取 ‘a’ 轴主编码器值
_ITa	读取 ‘a’ 轴独立运动时间常数值
_KSa	读取 ‘a’ 轴步进电机平滑常数值
_MTa	读取 ‘a’ 轴电机类型值
_RPa	读取 ‘a’ 轴由轮廓分配器产生的命令位置值
_TDa	读取 ‘a’ 轴步进计数寄存器值
_TPa	读取 ‘a’ 轴主编码器值

11. 双闭环（辅助编码器）

DMC-2x00 为每个轴的第二编码器提供了接口，但对配置为步进电机工作方式及用于输出比较的轴来说，第二编码器口不起作用。使用时，第二编码器一般都安装在电机或负载侧，但可以安装在任何位置。第二编码器最通常的用途是间隔补偿，如下所述。

第二编码器可以是标准正交型，也可以是脉冲+方向型。控制器也提供有转换编码器旋转方向的功能设定。用 CE 命令配置主、辅编码器。命令形式为 CE a, b, c, d, (或 a, b, c, d, e, f, g, h 用于 4 轴以上控制器)，其中，参数 a, b, c, d 各等于两个整数 m、n 之和。m 配置主编码器，而 n 配置辅助编码器。

注意：此项操作不能用于配置为步进电机的轴。

使用 CE 命令

m=	主编码器	n=	第二编码器
0	普通正交	0	普通正交
1	脉冲&方向	4	脉冲&方向
2	反向正交	8	反向正交
3	反向脉冲&方向	12	反向脉冲&方向

例如，要将主编码器配置为反向正交，m=2，而第二编码器为脉冲&方向，n=4，总和为 6，因此用于 A 轴的命令是：

CE6

11. 1 用于辅助编码器的其它命令

命令 DE a, b, c, d 用来定义辅助编码器的值，例如，

DE 0, 500, -30, 300

此命令将 A, B, C, D 轴辅助编码器的当前初始值定义为 0, 500, -30, 300。

辅助编码器的值可以用命令 DE? 进行查询，例如，

ED? , , ?

此命令返回 A, C 轴辅助编码器值。

辅助编码器位置可以用命令分配给变量，如下：

V1=_DEA

命令 TD a, b, c, d 返回辅助编码器的当前位置。

命令 DV a, b, c, d 将使用的辅助编码器配置成间隙补偿。

11. 2 间隙补偿

有两种使用辅助编码器的间隙补偿方法：

1. 连续双闭环
2. 采样双闭环

为了方便起见，我们看看电机和负载之间的联轴器有间隙的情况。为了补偿间隙，将位置编码器安装在电机和负载侧。

第一种方法，连续双闭环把两个反馈信号相结合以达到稳定，这种方法需要担心系统调整，且取决于间隙的大小。不过，一旦成功，这种方法就连续补偿间隙。

第二种方法，采样双闭环，只在终点读取负载编码器并进行修正。这种方法与间隙大小无关。不过它只对终点需要位置精度的点——点运动系统有效。

注意：还有一种间隙补偿的方法，那就是用测量仪器精确测出电机与负载之间的间隙值，然后将此值作为间隙补偿量输入到控制器中，在运动过程中予以补偿。中宝伦公司针对中国用户的实际情况，对某些型号的控制器的开发了此种补偿功能。但需要用户在订购时特别说明，中宝伦公司会额外收取一定费用。

11. 3 举例

举例1 连续双闭环

将负载编码器连接到主编码器口，将电机侧编码器连接到第二编码器口。双闭环方法把两个编码器之间的滤波器功能进行分离。以负载编码器为基础，将 **KP** 和 **KI** 用于位置误差补偿，将 **KD** 用于电机侧编码器。这种方法的系统稳定性很好。

用指令 **DV**（双速度环）使双闭环方法有效，其中：

DV1, 1, 1, 1

此命令就使 4 个轴的双闭环功能有效。

DV0, 0, 0, 0

此命令就使双闭环功能无效。

注意：双闭环补偿取决于间隙幅度大小，间隙过大，会使闭环系统不稳定。所建议的补偿步骤是以 **KP=0, KI=0** 开始，在 **DV1** 条件下，使 **KD** 值调至最大。一经求得 **KD**，就渐渐地将 **KP** 增至最大值，最后，如果需要，再增大 **KI**。

举例2 采样双闭环

在此例中，我们看看由旋转电机经滚珠丝杠带动直线滑台。由于滚珠丝杠有间隙，就需要用直线编码器（即光栅）来检测滑台位置；从稳定性方面来讲，在电机侧最好使用旋转编码器。将旋转编码器连接到 A 轴主编码器口，而将光栅连接到 A 轴的辅助编码器口。假定：所要的运动距离是 1 吋，它对应于旋转编码器反馈值为 40000cts，光栅反馈值为 10000cts。

设计方法是驱动电机移动一段距离，对应 40000cts 旋转编码器计数值，运动一完成，控制器就检测到光栅反馈的位移值，并进行位置修正。

由以下程序来实现：

指 令	说 明
# DUALOOP	标号
CE0	配置编码器
DE0	设置初始值
PR40000	主要运动
BG A	开始运动
# CORRECT	修正循环
AMA	等待运动完成

V1=10000-_DEA	求直线编码器误差
V2=_TEA/4+V1	补偿电机误差
JP# END, @ABS[V2]<2	若误差 <2, 就退出
PR V2*4	修正移动量
BGA	开始修正
JP# CORRECT	重复循环
# END	标号
EN	程序结束

12. 运动平滑（S 曲线加减速）

DMC-2x00 控制器具有速度轮廓平滑功能，通常也称之为 S 曲线加、减速功能。以此来减小系统的机械振动。

梯形速度轮廓（即线性加减速）使加速度突然从 0 变到设定的最大值，不连续的加速度往往会引起振荡，产生冲击。加速度轮廓平滑功能产生连续加速轮廓并减小机械冲击和振荡。

12. 1 用 IT 和 VT 命令使运动平滑（用于伺服电机）：

当使用伺服电机工作方式时，用 IT 和 VT 命令来实现运动平滑功能。这些命令对加、减速函数进行滤波，以产生平滑的速度轮廓。所产生的速度轮廓具有连续加速性能，从而减小机械振荡。

用以下命令来指定平滑功能：

IT a, b, c, d	独立运动时间常数
VTn	矢量运动时间常数

命令 IT 用于平滑 JG、PR、PA 等非插补方式的运动，而命令 VT 用于平滑 VM、LM 等插补方式的矢量运动。

平滑参数 a, b, c, d 和 n 是 0~1 之间的数，它决定滤波强度。最大值 1 不用于产生梯形速度轮廓的滤波器。平滑参数值越小意味着滤波作用越强，运动越平滑。

下例说明平滑效果，图 6.9 表示梯形速度轮廓及加入平滑滤波后的加速度和速度曲线。

注意：平滑处理使运动时间延长。

12. 2 举例

指 令	说 明
PR20000	位置
AC100000	加速度
DC100000	减速度
SP5000	速度
IT 0.5	平滑滤波器
BGA	开始运动

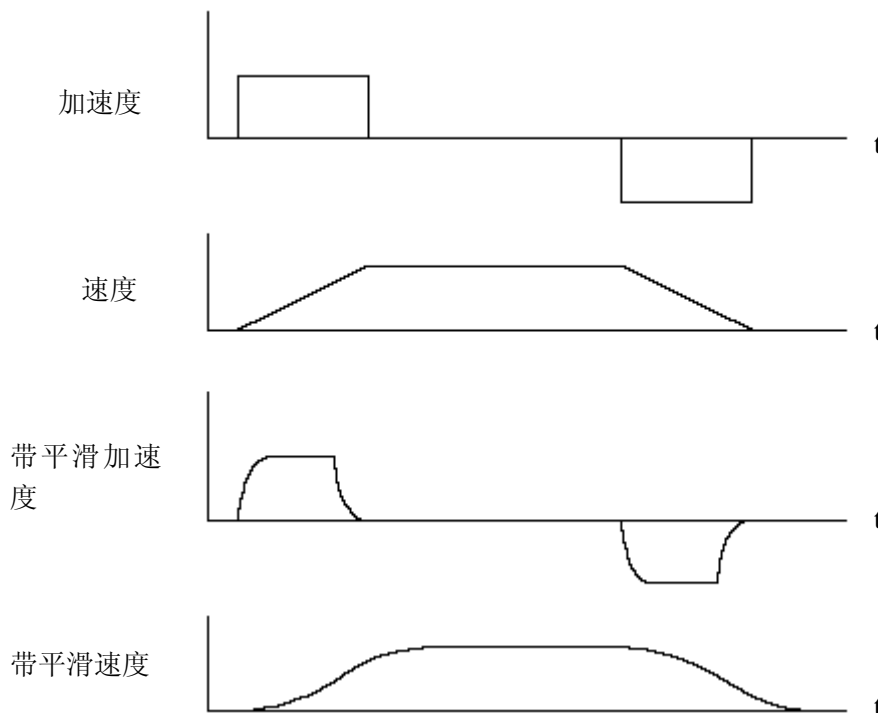


图 6.9- 梯形包络速度轮廓和平滑速度轮廓

12. 3 用 KS 命令使运动平滑（用于步进电机）：

当以步进电机方式工作时，用命令 **KS** 能实现运动平滑功能。**KS** 命令对步进电机脉冲的频率进行平滑处理。与命令 **IT**、**VT** 相类似，**KS** 也产生平滑的速度轮廓。

用以下命令指定步进电机平滑功能：

KS a, b, c, d

其中：a, b, c, d 是 0.5~8 之间的整数，它代表平滑量（滤波深度）。最小值 0.5 意味着没有滤波；仍然是梯形速度轮廓。滤波参数值越大，意味着滤波作用越强，运动就越平滑。

注意：**KS** 只对步进电机有效。

13. 原点返回功能

可以用 **FE**、**HM** 指令使电机回到机械参考点，此参考点称之为原点开关，连接到原点输入端子。**HM** 命令除检测原点输入信号之外，还检测电机编码器标志脉冲（通常称为一转脉冲）信号。用命令 **CN** 来定义原点输入信号的极性。

FE 命令对于检测原点开关很有用，原点开关连接到原点输入端。当使用 **FE** 和 **BG** 命令时，电机就会加速到指定的旋转速度，并旋转直至检测到原点开关输入信号状态变化，然后，电机就会减速停止。为了使电机在检测到原点开关之后快速减速，在执行 **FE** 命令之前必须输入大的减速值。产生的速度轮廓曲线示于图 6.10。

HM 命令能用来使控制器检测到原点头之后，命令电机按标志脉冲进行定位，从而获得更为精确的初始位置。**HM** 和 **BG** 命令的动作过程如下：

1. 一开始运动，电机加速到旋转速度，运动方向由原点输入的状态来决定。低电平使电机以正向开始运动，高电平使电机以反方向开始运动。用 **CN** 命令来定义原点输入极性。
2. 一检测到原点开关改变状态，电机就开始减速停止。

3. 然后电机就慢慢返回运转，直到再次检测到原点开关状态翻转。

4. 接着，电机正向运动，直至检测编码器标志脉冲。

5. DMC-2x00 将检测到标志脉冲的位置定义为原点位置 (0)。

关于其它几种原点返回方法及原点返回程序的设计技巧请参考本书第 3 章中关于“原点开关输入”一节的说明及第二篇命令手册中关于 HM、FE、FI 命令解释。

13. 1 举例

指 令	说 明
# HDME	标号
AC1000000	加速度
DC1000000	减速度
SP5000	回原点速度
HMA	A 轴原点返回
BGA	开始运动
AMA	运动完成后
MG “AT HOME”	发送信息
EN	结束
# HDGE	标号
AC2000000	加速度
DC2000000	减速度
SP8000	速度
FE B	搜寻原点开关边缘命令
BG B	开始运动
AMB	运动完成后
MG “FOUND HOME”	发送信息
DP, 0	定义此位置为 B 轴 0 点
EN	结束



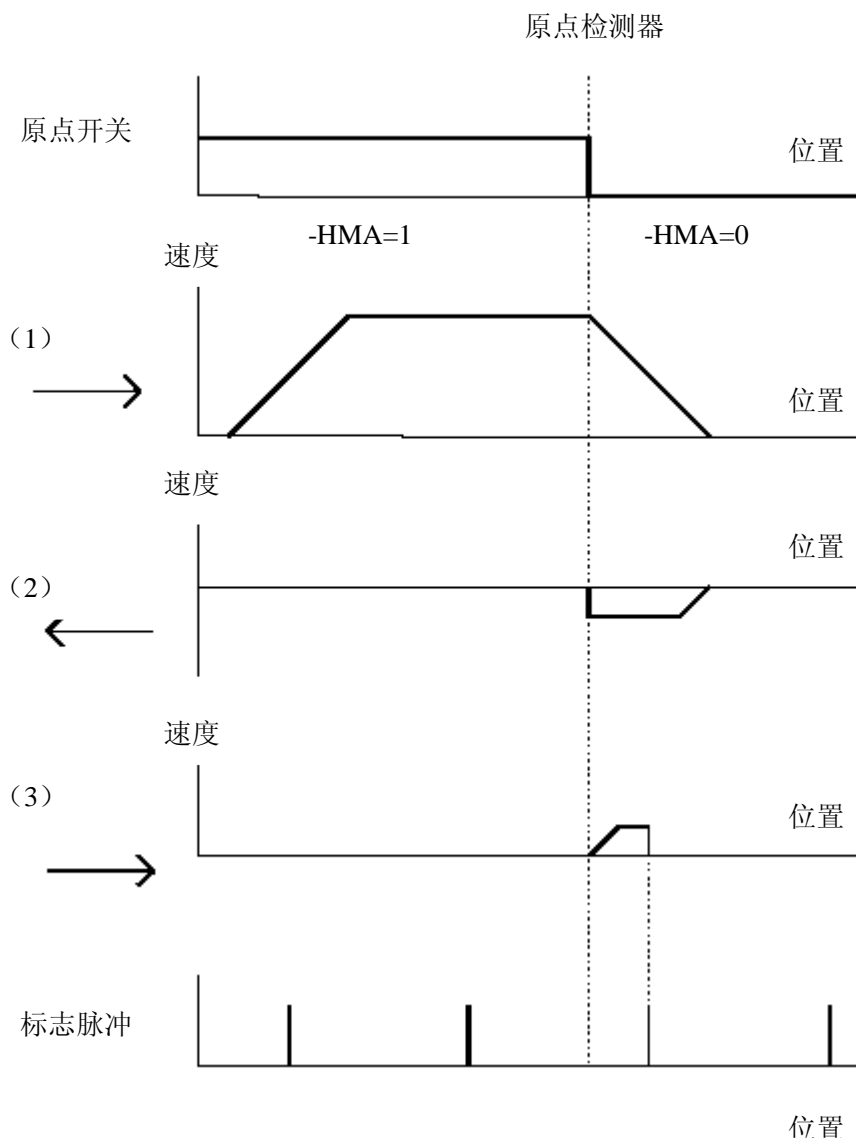


图 6.10- 原点返回动作顺序

图注：(1) 图表示电机开始使工作台朝原点开关方向运动
 (2) 图表示电机驱动工作台朝原点开关反方向运动
 (3) 图表示电机驱动工作台朝标志脉冲方向运动

13. 2 原点返回命令汇总

命 令	说 明
FE ABCD	搜寻原点开关边缘，此命令检测原点开关输入信号
FI ABCD	搜寻标志脉冲，此命令检测标志脉冲输入信号
HM ABCD	原点返回命令，此命令将 FE、FI 命令结合使用
SC ABCD	停止代码
TS ABCD	查询开关和输入状态

13. 3 原点返回操作数汇总

操 作 数	说 明
-------	-----

_HM a	读取原点输入开关状态值
_SC a	读取停止代码
_TS a	读取开关和输入状态

14. 高速位置捕获（位置锁存功能）

对于印刷套色等需要定标应用来说，往往需要精密捕获位置。DMC-2x00 具有位置锁存功能，当锁存输入状态发生变化时，用此功能就能够捕获到各轴主编码器或辅助编码器的位置。对此功能进行配置，从而当锁存输入变高或变低时对位置进行捕获。当锁存功能使能为有效低电平工作时，在 12 μ sec 之内就会捕获到位置值，当锁存使能为有效高电平工作时，在 35 μ sec 之内就会捕获到位置值。各轴都有一个为位置捕获而对应的通用输入；如下：

输 入	功 能	输 入	功 能
IN1	A 轴锁存	IN9	E 轴锁存
IN2	B 轴锁存	IN10	F 轴锁存
IN3	C 轴锁存	IN11	G 轴锁存
IN4	D 轴锁存	IN12	H 轴锁存

用 DMC-2x00 软件命令 AL, RL 使锁存功能使能，并给出锁存的位置。使用锁存的步骤如下：

1. 给出 ALABCD 命令使锁存对主编码器使能，而 AL SASBSCSD 对辅助编码器锁存使能；
2. 用 _ALA 或 B 或 C 或 D 命令，测试看看锁存是否出现（输入变低电平）。例如，V1=_ALA 将轴 A 锁存的状态放入 V1 变量，如果未出现锁存，V1=1；
3. 在出现锁存之后，用 RLABCD 或 _RLABCE 读取捕获的位置。

注意：在每次锁存发生之后，必须对锁存重新使能。

举例：

指 令	说 明
# LATCH	标号
JG, 5000	JOG B 轴
BGB	B 轴开始运动
ALB	B 轴锁存使能
# WAIT	等待循环程序标号
JP# WAIT, _ALB=1	如果锁存未出现，重复循环等待
Result=_RLB	设置 Result 等于 B 轴锁存的位置
Result=	打印结果
EN	程序结束

第七章 应 用 编 程

1. 概述

在第 6 章中，我们对 DMC-2x00 所具有的基本运动控制功能做了全面介绍。以此为基础，用户即可以利用 API 函数用 VB、VC、C/C++、Labview、delph 等高级语言编写满足特殊需要的应用程序，也可以利用控制器已具有的运动控制功能编写这些应用程序。一般来说，利用已有的运

动控制功能就可编写出满足特殊需要的应用程序；只不过在某些应用场合，需要具有良好的人机界面，在此情况下，用户有必要采用高级语言进行编程。

我们已知道，DMC-2x00 提供了功能强大、简单易记的编程语言，使用户能够精心打造满足特殊应用要求的、具有自身特点的专用控制器。本章以前 6 章内容为基础，全面介绍创建应用编程的方法和技巧，使用户能够灵活而充分地运用 DMC-2x00 的基本功能。

GALIL 运动控制器本身具有 32bit 高速处理器及 Flash、RAM 存储器，进行程序、命令解释、运动模式规划、I/O 事件处理、高速插补运算、闭环控制、通信管理等。因此我们可以把程序下载到 DMC-2x00 存储器中，然后脱离主机运行，主机可以从事其它任务处理。不过，在需要的情况下，主计算机仍然能够随时给控制器发送命令，即使在程序正在执行当中，也是如此。对于这种应用编程而言，只能使已提供的 ASCII 格式的命令。

除标准运动命令之外，DMC-2x00 还提供了一些命令，使 DMC-2x00 能根据运行中的一些条件进行自身判断。这些命令包括条件跳转、事件触发及子程序等。例如，命令 JP# LOOP, n <10, 如果变量 n <10, 程序就跳转到标号# LOOP。

为使编程更为灵活，DMC-2x00 具有用户自定义变量、数组及算术运算功能。例如，在定长裁断应用中，在程序中，能够把长度指定为变量，从而操作者就可以根据现场需要随时加以改变。

用户程序存储区大小为 10000 行× 80 字符。

2. 程序编辑

GALIL 提供了两种版本的编辑器，一种是 DOS 版的内编辑器，另一种是 Windows 版本，使用户能够很方便地在控制器的存储器中创建并编辑程序。在 DOS 环境下，内编辑器用命令 ED 打开；而在 Windows 环境下，只要运行 SmartTerm 文件，就会自动打开基于编辑器的 Windows。基于编辑器的 Windows 具有更多的功能，而且容易使用，因此，建议用户尽可能使用基于编辑器的 Windows，而当使用简单终端时，才使用内编辑器。

一给出 ED 命令，各程序行就自动按序从 000 开始编号。如果没有参数跟随在 ED 命令之后，编辑器提示符就会对存储器中的上一个程序的上行缺省。如果需要的话，通过在 ED 后面指定行号或标号，用户就能对指定的行号或标号进行编辑。这种内编辑器几乎与 DOS 版中的行编辑器软件相同。

注意：ED 命令只接受 DOS 窗口中的参数（如# BEGIN）。为了通用起见，本节中的编辑功能，不是 DOS 方式时就不能适用。

指 令	说 明
: ED	将编辑器放到上个程序的结尾处
: ED5	将编辑器放到第 5 行
: ED #BEGIN	将编辑器放到标号# BEGIN 处

行号表示为 000, 001.002 以此类推，跟着行号输入程序命令。一行中可以编辑多个命令，只要总字符不要超过 80 个即可。

在编辑方式下，为了存储、插入、删除程序行，程序员可以调用特殊指令。这些特殊指令列表如下：

2.1 编辑方式命令

〈RETURN〉 回车命令

键入回车键使输入指令的当前行保存起来，编辑器会自动转入下一行。因此，提示许多〈RETURN〉会使编辑器前进许多行。注意，如果不给〈RETURN〉，就不会保存程序行的更改。

〈Cntrl〉 p

〈Ctrl〉p 命令使编辑器移到前一行。

〈Ctrl〉I

〈Ctrl〉I 命令在当前行之前插入一行。例如，如果编辑器处于第 2 行，并用 〈Ctrl〉I 命令，就会将新行插入第 1 行和第 2 行之间，新行的行号为 2，原第 2 行变为第 3 行。

〈Ctrl〉D

〈Ctrl〉D 删除当前正在编辑的行。例如，如果编辑器处在第 2 行。用 〈Ctrl〉D 就会删除该行，原第 3 行又变为第 2 行。

〈Ctrl〉Q

〈Ctrl〉Q 退出编辑方式，DMC-2000 会返回一个冒号 (:) 作为回应。

在编辑完结束后，用户可以对编写的程序用 LS 命令进行列表，如果没有参数跟随在 LS 命令之后，就会列出整个程序。用户可以用参数 n 对所指定的行或标号进行列表。跟在起始列表参数之后的命令和新行号或标号指定列表要停止的位置。

2. 2 举例：

指 令	说 明
: LS	对整个程序列表
: LS5	在第 5 行开始列表
: LS5, 9	列出 5~9 行
: LS #A, 9	列出#A~9 行
: LS #A, #A+5	列出标号#A 和其余 5 行

注意：这种编辑器不适用于 DMC-2100，不过，可以使用任何其它终端（如 Telnet）。

3. 程序格式

DMC 程序由配合解决机械设备控制要求的 DMC-2x00 指令组成，如起动、停止运动等动作性指令，与程序流程指令相结合来形成完整的程序。程序流程指令对实时条件，如延迟时间或运动完成（到位）等进行评估判断，并以此改变程序流程。程序中的每条指令必须用分号 (;) 分开或用回车键结束。用分号把一行中的多条指令分开，每行中最多不能超过 80 个字符。用回车键使程序行结束。

3. 1 在程序中使用标号

所有 DMC-2x00 程序必须以标号开始，以 EN 语句结束。标号以 # 号开始，后面最多可跟随 7 个字符。第一个字符必须是字母，其后，允许使用数字，但不允许有空格。

对于 1.0c 以上版本的控制软件，最多可以定义 510 个标号。

有效标号举例：

```
# BEGIN
# SQUARE
# X1
# BEGIN1
无效标号举例：
#1 Square
#123
```

举例：

指 令	说 明
-----	-----

# START	程序标号
PR10000, 20000	指定 A、B 轴相对运动距离
BG AB	开始运动
AM	等待运动完成
WT2000	等待 2sec
JP# START	跳转到标号 START
EN	程序结束

上述程序使 A、B 两轴分别移动 10000 和 20000 计数单位，在运动完成后，电机停 2sec，循环无限重复直到发送停止命令。

3. 2 特殊标号

DMC-2x00 有一些特殊标号，以此来定义输入中断子程序、限位开关处理子程序、报警处理子程序、命令出错子程序。参见“自动启动子程序”一节。

DMC-2x00 有一个特殊标号用于自动程序执行。在上电或复位时，就自动执行已存入控制器非易失性存储器并以标号# AUTO 开始的程序。必须用命令 BP 将程序存入非易失性存储器。

对条件进行监测的自动子程序见后续说明。

指 令	说 明
# ININT	输入中断子程序标号
# LIMSWI	限位开关处理子程序标号
# POSERR	存取位置误差子程序标号
# MCTIME	运动完成条件启动超时子程序标号
# CMDERR	命令出错子程序标号
# COMINT	辅助串口通信中断子程序
# TCPERR	TCP/IP 通信出错子程序标号（只用于 DMC2100/2200）

3. 3 注释程序

注释程序有两种方法，第一种方法使用 NO 命令，使程序籍入 GALIL 程序；第二种方法是使用 REM 语句，这需要使用 GALIL 软件来进行。

3. 3. 1 NO 命令

DMC 控制器提供 NO 命令用于对程序注释。在 NO 命令之后，一行中可以写入 78 个字符，让程序员对程序进行注释，举例如下：

指 令	说 明
# PATH	标号
NO2-D CIRCULAR PATH	注释——无操作
VM AB	矢量方式
NO VECTOR MOTION ON A AND B	注释——无操作
VS 10000	矢量速度
NO VECTOR SPEED S 10000	注释——无操作
VP -4000, 0	矢量位置
NO BOTTOM LINE	注释——无操作
CR 1500, 270, -180	圆弧运动
NO HALF CIRCLE MOTION	注释——无操作
VP 0, 3000	矢量位置

NO TOP LINE	注释——无操作
CR 1500, 90, -180	圆弧
NO HALF CIRCLE MOTION	注释——无操作
VE	矢量结束
NO END VECTOR SEQUENCE	注释——无操作
BGS	开始运动
NO BEGIN SEQUENCE MOTION	注释——无操作
EN	结束程序
NO END OF PROGRAM	注释——无操作

注意：NO 命令是实际控制器命令，因此，NO 命令会占用控制器的处理时间。

提示：有些用户用字“NOTE”对程序加注，在“NO”之后的每个字均视为注释。

3. 3. 2 REM 命令

如果您使用 GALIL 软件与 DMC-2x00 控制器通信，就可以用 REM 语句。‘REM’ 语句以字‘REM’ 开始，在同一行中可以跟随任何注释。当下载程序到控制器时，GALIL 终端软件会去掉这些语句。例如：

```
# PATH
REM 2-D CIRCULAR PATH
VMAB
REM VECTOR MOTION ON A AND B
VS 10000
REM VECTOR SPEED IS 10000
VP -4000, 0
REM BOTTOM LINE
CR 1500, 270, -180
REM HALF CIRCLE MOTION
VP 0, 3000
REM TOP LINE
CR 1500, 90, -180
REM HALF CIRCLE MOTION
VE
REM END VECTOR SEQUENCE
BGS
REM BEGIN SEQUENCE MOTION
EN
REM END OF PROGRAM
```

当下载此程序到控制器时，会去掉这些 REM 语句。

4. 执行程序——多任务

DMC-2x00 具有 8 个通道，同时运行 8 个独立程序。把这些程序称之为通道。分别从 0~7 进行编号，其中 0 是主通道。多任务功能对于执行功能相对独立的操作，如 PLC 功能等非常有用。

主通道与其它通道在以下方面有所不同：

- (1) 只有主通道（0 通道）可以使用输入命令 IN
- (2) 限位开关处理、位置超差、命令出错等子程序均在 0 通道执行。

要开始执行程序，使用如下指令：

XQ # A, n

其中: n 表示通道号, 要暂停执行任意通道, 用指令:

HX n

其中: n 是通道号

注意: XQ 和 HX 命令能通过运行中的程序来执行。

下例是在输出点 1 端产生波形的独立运动程序。

指 令	说 明
# TASK 1	TASK 1 标号
AT0	初始化参考时间
CB1	清除输入点 1
# LOOP1	LOOP1 标号
AT 10	从参考时间开始等待 10ms
SB1	输出点 1 置高电平
AT -40	从参考时间开始等待 40ms, 然后初始化参考时间
CB1	清除输出点 1
JP# LOOP1	重复 LOOP1
# TASK2	TASK2 标号
XQ # TASK1, 1	执行 TASK1
# LOOP2	LOOP2 标号
PR 100	定义相对距离
BGX	开始运动
AMX	在运动完成后
WT10	等待 10ms
JP# LOOP2, @IN[2]	输入点 2 不变低, 就一直重复运动
HX	暂停所有任务

用 XQ#TASK2, 0 指令执行上述程序, TASK2 在主通道 (即通道 0), 在 TASK2 之内执行 #TASK1。

5. 调试程序

DMC-2x00 提供了用于调试应用程序的命令和操作数。这些命令有助于监测程序执行, 确认控制器状态和控制器程序、数组和变量空间大小的查询命令。操作数也包含能够有助于调试程序的重要状态信息。

5.1 跟踪命令 (适用于 DMC-2100/2200)

跟踪命令会在程序执行之前, 使控制器将程序中的每一行立即发送到主计算机。用命令 TR1 使跟踪功能使能, TR0 使跟踪功能关断。

从控制器送出的数据存储在输出 FIFO 缓冲区中。输出 FIFO 缓冲区能存储 512 个字符信息。在正常运行中, 控制器把输出内容放入 FIFO 缓冲区, 主计算机侧的软件监测缓冲区并读取所需信息。当跟踪方式使能时, 控制器就以非常高的速率把这些信息发送到 FIFO 缓冲区。一般来说, 由于软件读取信息较慢, 因此 FIFO 就会放满, 当 FIFO 放满时, 如不清除, 就会使程序执行速度减慢。如果用户想避免这种不必要的延迟, 就发送命令 CW, 1, 此命令使控制器把那些不能放入 FIFO 的数据扔掉。在此情况下, 控制器就没有延迟程序执行。

5.2 错误代码命令

当有程序错误时，DMC-2x00 就把程序执行暂停在错误发生点。要显示程序执行的上一行号，就用命令 MG_ED。

用户用命令 TC1 能得到错误发生条件的类型，此命令报告错误代码号和描述出错条件的文本信息，命令 TC0 或 TC 返回不带文本信息的错误代码。详见第二篇命令手册中有关 TC 命令的说明。

5.3 停止代码命令

用停止代码命令 SC 能确认各轴的运动状态，当某轴运动莫名其妙停止时，此命令很有用。SC 命令会返回代表运动状态的代码号。详见第二篇命令手册。

5.4 RAM 存储器查询命令

为了调试程序存储器，数组存储器或变量存储器的状态，DMC-2x00 有几个很有用的命令。命令 DM? 返回当前可用的数组元素数；命令 DA? 返回当前能被定义的数组数。例如，一台标准 DMC-2x10 最多有 30 个数组，8000 个数组元素。如果已定义了一个 100 个元素的数组，DM? 命令返回值就为 7900，DA? 命令返回值就为 29。

为了列出变量区的内容，使用查询命令 LV（变量列表）即可；为了列出数组区中的内容，就使用查询命令 LA（数组列表）；为了列出程序区的内容，使用查询命令 LS（列表）；为了只列出应用程序标号，就使用查询命令 LL（标号列表）。

5.5 操作数

一般来说，所有操作数均提供在调试应用程序中有用的相关信息。下面列出的这些操作数对程序调试特别有用。为了显示操作数的值，可以使用信息命令。例如，由于操作数_ED 读取程序执行的最后一行，那么，命令 MG_ED 就会显示此行号。

_ED 读取程序执行的最后一行。用于确定程序停止的地方。

_DL 读取可用的标号数

_UL 读取可用的变量数

_DA 读取可用的数组数

_DM 读取可用的数组元素数

_AB 读取可用的急停输入状态

_FLa 读取可用的‘a’轴正向限位开关的状态

_RLa 读取可用的‘a’轴负向限位开关的状态

5.6 举例

下例有一个错误，在 A 轴处于运动中时，试图指定新的相对位移量。当执行此程序时，控制器停在 003 行，然后，用户就可以用命令 TE1 对控制器发问查询，控制器就会以相对应的说明给予回应：

指 令	说 明
: ED	编辑方式
000# A	标号
001 PR1000	相对位置 1000
002 BGA	开始
003 PR5000	相对位移 5000
004 EN	END

〈Ctrl〉 Q	退出编辑方式
: XQ#A	执行#A
? 003 PR5000	第 3 行有错误
: TC1	查询错误代码
? 7 Command not valid while running	在运行当中, 命令无效
: ED3	编辑第 3 行
003 AMX; PR5000; BGA	在运动完成后加入
〈Ctrl〉 Q	退出编辑方式
: XQ# A	执行#A

6. 程序流程命令

DMC-2x00 提供了控制程序流程的命令, DMC-2x00 通常按序执行程序指令。使用事件触发器、条件启动和条件跳转语句就能改变程序流程。

6. 1 事件触发器和条件启动

要使控制器离开主计算机独立运行, 就要对 DMC-2x00 进行编程, 根据出现事件做出决策。这样的事件有如下几种; 等待运动完成、等待定时到达、等待输入逻辑电平改变。

DMC-2x00 提供了好几种事件触发器, 使程序暂停, 直到所指定的事件出现才会继续程序执行。通常, 自动按序逐行执行程序。不过, 对事件触发指令译码时, 暂停实际程序正常顺序, 如果事件触发器触发, 程序就不会按序继续执行。例如, 能够用运动完成触发器把程序中的两个运动顺序分开; 直到运动按第一个运动顺序完成才会执行用于第二个运动顺序的命令。在这种方式, 没有主计算机干预, DMC-2x00 也能按照自身状态或外部事件进行决策。

注意: 对于 DMC-2100/2200 不推荐这种方式。当使用主机和控制器之间的通信暂停的事件触发时, 缓冲区容易堆积。

6. 1. 1 事件触发器

DMC-2x00 事件触发器

命 令	功 能
AM ABCDEFGH 或 S	直到所指定轴或运动按序完成程序才继续执行。不带参数的 AM 适用于所有轴的运动完成, 此命令对分开程序中的运动顺序很有用。
AD A 或 B 或 C 或 D 或 E 或 F 或 G 或 H	使程序执行暂停, 直到位置命令到达所指定的相对于运动始点的距离。一次只能指定一个轴
AR A 或 B 或 C 或 D 或 E 或 F 或 G 或 H	使程序执行暂停, 直到由上一个 AR 或 AD 命令所指定的距离过后, 才执行后续命令。一次只能指定一个轴。
AP A 或 B 或 C 或 D 或 E 或 F 或 G 或 H	使程序执行暂停, 直到绝对位置出现之后, 才继续执行。每次只能指定一个轴。
MFA 或 B 或 C 或 D 或 E 或 F 或 G 或 H	使程序执行暂停, 直到正向运动到达绝对位置后才继续执行。一次只能指定一个轴。如果位置已经越过指定位置点, MF 会立即释放。此功能对齿轮轴或辅助编码器输入起作用。
MR A 或 B 或 C 或 D 或 E 或 F 或 G 或 H	使程序执行暂停, 直到反向运动到达绝对位置之后才继续执行。一次指定一个轴。如果位置已越过指定位置点, MR 会立即释放。此功能对齿轮轴或辅助编码器输入起作用。
MCA 或 B 或 C 或 D 或 E 或 F 或 G 或 H	使程序执行暂停, 直到运动轮廓已经完成且编码器实际位置进入或越过指定位置才继续执行。如果不到位, TWA, B, C, D 设置超时报

	警。如果出现超时，条件启动就会清除，并将停止码设置为 99。应用程序会跳转到标号#MCTIME。
AI±n	使程序执行暂停，直到所指定的输入处于所规定的逻辑电平，n 指定输入点。正为高电平，负为低电平。n=1~8（用 DMC-2x10~2x40 时），n=1~24（用 DMC-2x50~2x80 时）n=1~80（用 DMC-2x80 时）
AS ABCDEFGH	使程序执行暂停，直到所指定轴到达其旋转速度
AT±n	使程序执行暂停，直到从起始计时 n msec 到达才继续执行。AT0 设置参考始点，ATn 从参考始点开始等待 n msec。AT-n 从参考始点开始等待 n ms 并在时间到达之后设置新的参考始点。
AVn	使程序执行暂停，直到沿联动轨迹指定的距离出现
WTn	使程序执行暂停，直到所指定的等待时间到达，以ms 为单位。

6. 1. 2 举例

举例1 多重运动顺序

用 AM 条件启动命令把两个 PR 运动分开，如果不使用 AM 命令，由于只有在运动完成后才指定新 PR 值，所以对第二个 PR 命令，控制器会返回问号（？）

指 令	说 明
# TWOMOVE	标号
PR 2000	相对位置值 2000
BGA	开始运动
AMA	等待运动完
PR 4000	下一位置运动
BGA	开始第二个运动
EN	程序结束

举例2 在距离之后设置输出

在离开运动始点 1000cts 距离之后设置输出位 1 为高电平。条件启动的精度是采样周期倍乘的速度。

指 令	说 明
# SETBIT	标号
SP10000	速度为 10000
PA 2000	绝对位置
BGA	开始运动
AD 1000	等待 1000cts 到达
SB1	设置输出位 1
EN	程序结束

举例3 重复性位置触发

在运动期间，每 10000cts 设置输出位，用 AR 条件启动。

指 令	说 明
# TRIP	标号
JG 5000	指定 JOG 速度
BGA; n=0	开始运动
# REPEAT	重复循环#REPEAT
AR 10000	等待 10000cts 到达
TPA	报告位置

SB1	设置输出 1
WT50	等待 50ms
CB1	清除输出 1
n=n+1	循环计数+1
JP# REPEAT, n<5	重复循环 5 次
STA	停止
EN	程序结束

举例 4 按外部输入开始运动

此例等待输入点 1 变低，然后开始运动

注意：AI 命令实际上使程序执行暂停，只有当输入出现时，才继续执行。如果您不想使程序执行顺序暂停，那么，您就要使用输入中断功能（II）或条件跳转功能如：JP# GO, @IN[1]=-1。

指 令	说 明
# INPUT	程序标号
AI-1	等待输入点 1 变低
PR10000	相对位置 10000
BGA	开始运动
EN	程序结束

举例 5 当速度到达时，设置输出点

指 令	说 明
#ATSPEED	程序标号
JP50000	指定 JOG 速度
AC 10000	加速度
BGA	开始运动
ASA	等待速度到达 50000
SB1	输出点 1 置高
EN	程序结束

举例 6 改变矢量轨迹速度

下列程序使进给率或矢量速度在指定的矢量运动距离点发生改变。矢量距离从运动始点或上一个 AV 命令开始计算：

指 令	说 明
# VECTOR	标号
VMAB; VS5000	联动轨迹
VP10000, 20000	矢量位置
VP20000, 30000	矢量位置
VE	矢量结束
BGS	开始运动
AV5000	在矢量距离 5000 到达之后
VS1000	减速
EN	结束

举例 7 带有等待的多重运动

本例通过在执行新的运动之前等待各个运动完成进行多重相对位移运动：

指 令	说 明
# MOVES	标号
PR12000	距离

SP20000	速度
AC100000	加速度
BGA	开始运动
AD 1000	等待 10000cts 距离到达
SP5000	新速度
AMA	等待运动完成
WT200	暂停 200ms
PR-10000	新位置
SP30000	新速度
AC150000	新加速度
BGA	开始运动
EN	结束

举例 8 用 AT 定义输出波形

下列程序在输出点 1 产生 10ms 高电平，40ms 低电平的脉冲列，每个循环每 50ms 重复一次。

指 令	说 明
# OUTPUT	程序标号
AT0	初始化时间参考点
SB1	输出点 1 置高
# LOOP	循环
AT10	从参考点计时 10ms 之后
CB1	输出点 1 置低
AT40	从参考点计时等待 40ms 并重新设置参考点
SB1	输出点 1 置高
JP# LOOP	循环
EN	结束

6. 2 条件跳转

DMC-2x00 具有条件跳转和条件跳转到子程序功能；分别对应命令 JP 和 JS，使程序根据所指定的条件跳转到新的分支程序中去执行。条件跳转确认条件是否满足，然后输入新位置或子程序。与事件触发不同，条件跳转命令不暂停程序。条件跳转功能对于实时检测事件很有用。此功能使 DMC-2x00 不用主计算机干预即可做出判断。例如，DMC-2x00 能以输入点状态为条件在两种运动轮廓之间做出选择。

6. 2. 1 命令格式——JP 和 JS

格 式	说 明
JS 目标地，逻辑条件	如果逻辑条件满足，跳转到子程序
JP 目标地，逻辑条件	如果逻辑条件满足，跳转到主程序某处

目标地是当指定条件满足时程序定序器将要跳转的程序行号或标号。注意：程序存储区的首行行号为 0。逗号意指向“IF”，逻辑条件通过逻辑算子测试两个操作数。

6. 2. 2 逻辑算子：

算 子	说 明
<	小于
>	大于

=	等于
<=	小于或等于
>=	大于或等于
<>	不等于

6. 2. 3 条件语句:

如果对不为 0 的任意值进行评估, 条件语句被满足。条件语句可以是任何有效的 DMC-2x00 数字操作数, 包括变量、数组元素、数值、函数、键字、算术表达式等。如果不给出条件语句, 即为无条件跳转。

数字	V1=6
数字表达式	V1=V7*6
	@ABS[V1] >10
数组元素	V1<Coun+[2]
变量	V1<V2
内部变量	_TPA=0
	_TVA>500
I/O	V1>@AN[2]
	@AN[1]=0

6. 2. 4 多重条件语句

DMC-2x00 会在一个跳转语句中接受多个条件。使用 “&” 和 “|” 操作数就能把多个条件语相迭加, 任意两个条件之间的 “&” 操作数要求两个语句都必须为真。任意两个条件之间的 “|” 操作数要求只要任何一个语句为真。

注意: 必须把各个条件放在圆括弧内, 以便控制器进行正确判断。此外, DMC-2x00 从左到右执行运算。关于数字表达式及位操作算子 ‘&’ 和 ‘|’ 的详细介绍, 请参见 “数字表达式” 一节。

例如, 使用以 V1、V2、V3、V4 命名的变量:

JP# TEST, (V1<V2) & (V3<V4)

在此例中, 若 V1 小于 V2 且 V3 小于 V4, 程序就跳转到标号为#TEST。为了进一步说明, 我们再看看下例:

JP# TEST, ((V1<V2) & (V3<V4)) | (V5<V6)

如果满足两个条件: 1. 如果 V1 小于 V2 且 V3 小于 V4, 或者 2. (V5<V6), 程序就会跳转到标号#TEST。

6. 2. 5 举例:

例 1 如果满足 JP 命令的条件, 控制器就跳转到所指定的标号或行号, 并从此点开始继续执行命令。如果条件不满足, 控制器就按序继续执行下一条命令。

指 令	说 明
JP # LOOP,count<10	如果变量count小于10, 跳转到# LOOP
JS # MOVE2,@IN[1]=1	如果输入点1为逻辑高电平, 跳转到子程序#MOVE2
	在执行完#MOVE2子程序之后, 程序分序器返回到调用子程序处的主程序位置。
JP # BLUE,@ABS[V2]>2	如果变量V2的绝对值大于2, 跳转到# BLUE
JP # C,V1*V7<=V8*V2	如果V1值乘V7小于或等于V8乘V2的值, 跳转到#C
JP #A	无条件跳转到#A

例2 使A轴电机运动到绝对位置1000cts, 并返回到0, 重复10次, 在每两次运动之间暂停100ms。

指 令	说 明
# BEGIN	程序标号
Count=10	循环计数器置初值
# LOOP	循环程序标号
PA1000	绝对位置1000
BGA	开始运动
AMA	等待运动完成
WT100	暂停100ms
PA0	返回绝对位置0点
BGA	开始运动
AMA	等待运动完成
WT100	等待100ms
Count= Count-1	循环计数器减1
JP# LOOP, Count>0	通过循环测试10次
EN	程序结束

6. 3 If、Else、End if

DMC-2x00提供了对使用IF、ELSE、ENDIF命令的条件语句的结构化方法。

6. 3. 1 使用IF、ENDIF命令

IF条件语句由IF和ENDIF命令组合来构成, IF命令具有一个或更多的条件语句作为其判断。如果条件语句判断为真, 命令解释器就会继续执行跟随在IF命令之后的命令; 如果条件语句判断为假, 控制器就会忽略掉在IF命令之后的命令, 去执行与之相配合的ENDIF命令或者去执行在程序中出现的ELSE命令 (见以下相关ELSE命令介绍)。

注意: 对于已执行的每个IF命令, 总是必须执行一次ENDIF命令。建议用户不要在IF条件语句里包含跳转命令, 否则会使程序执行不知去向。在此情况下, 命令解释器就可能不执行ENDIF命令。

6. 3. 2 使用ELSE命令

ELSE命令是IF条件语句的选择部分, 只有当IF命令的参数判断为假时才执行ELSE命令。ELSE命令必须出现在IF命令之后, 且没有参数。如果用于IF命令的参数判断为假, 控制器就会跳过后续命令去执行ELSE命令。如果用于IF命令的参数判断为真, 控制器就会执行IF和ELSE命令之间的命令。

6. 3. 3 箱套IF条件语句

DMC-2x00允许IF条件语句包含在其它IF语句之内, 此技术称之为‘箱套’, DMC-2x00允许255个IF条件语句进行箱套。这是一项功能非常强大的技术, 使用户能对许多种不同的分支情况进行指定。

命令格式 IF ELSE ENDIF

格 式	说 明
IF 条件语句	如果条件为真, 就执行IF命令后 (直到ELSE命令) 的命令, 否则, 继续在IF命令或可选的ELSE命令处执行。
ELSE	可选择命令, 当IF命令的参数判断为假时, 转而执行此命令, 只

	能与IF命令配合使用。
ENDIF	用来使IF条件语句结束的命令。对于每条IF命令，程序必须有一个ENDIF命令与之对应。

指 令	说 明
# TEST	开始主程序 “TEST”
II „3	输入点1和2时终端使能
MG “WAITING FOR INPUT1,INPUT2”	输出信息
# LOOP	无休止循环用标号
JP# LOOP	无休止循环
EN	主程序结束
# ININT	输入中断子程序
IF(@IN[1]=0)	以输入点1为基础的IF条件语句
IF(@IN[2]=0)	若第1个IF条件为真，执行第2个IF
MG “INPUT1 AND INPUT2 ARE ACTIVE”	若第2个IF条件为真，执行信息发送
ELSE	用于第2个语句的ELSE命令
MG “ONLY INPUT1 IS ACTIVE”	若第2个IF条件为假，执行信息发送
ENDIF	第2个条件语句结束
ELSE	用于第1个语句的ELSE命令
MG “ONLY INPUT2 IS ACTIVE”	若第1个IF条件为假，执行信息发送
ENDIF	第1个条件语句结束
# WAIT	用于循环的标号
JP# WAIT,(@IN[1]=0)(@IN[2]=0)	循环直到输入点1和2均无效
RI 0	中断返回

6. 4 子程序

所谓子程序就是一组以标号开始，且以EN命令结束的指令。子程序从主程序中用跳转子程序指令JS，后跟标号或行号和条件语句进行调用。可以嵌套8个子程序，在执行完子程序之后，程序定序器返回到调用子程序的地方。但子程序堆栈被变换的情况除外。

例如 画出边长为500cts正方形的子程序举例如下，正方形以矢量位置1000，1000处开始绘制。

指 令	说 明
# M	主程序标号
CB1	清除输出点1（抬笔）
VP1000, 1000; LE; BGS	定义矢量位置；移动笔
AMS	等待运动完成
SB1	输出点1置高电平（下笔）
JS# SQUARE; CB1	跳到SQUARE子程序
EN	结束子程序
# SQUARE	SQUARE子程序
V1=500; JS# L	定义边长
V1=-V1; JS# L	切换方向
EN	结束子程序
# L; PRV1, V1; BGA	定义A、B轴位置，开始A轴运动

AMA; BGB; AMB
EN

在A轴运动完成后，开始B轴运动
结束子程序

6.5 堆栈变换

可以用ZS命令使子程序堆栈进行变换。每次JS指令都执行一次中断或自动子程序（如# POSERR或# LIMSWI），子程序堆栈就加1。通常用EN指令存储堆栈。偶尔，希望不返回到子程序或中断被调用的程序行。ZS1命令使堆栈指针减1，这样就能使程序定序器继续执行下一行。ZS0命令把堆栈复位到它的初始值。例如，如果限位发生，执行# LIMSWI子程序，往往希望重新开始程序顺序，而取代返回到限位发生的地方。要满足这种要求，就在# LIMSWI子程序的末尾加上ZS0命令。

6.6 自动启动子程序

DMC控制器有一个特殊标号用于自动程序执行。已经存入控制器非易失存储器中以标号# AUTO开始的程序在上电或复位时能够自动被运行。此程序必须用BP命令存入非易失性存储器中。

6.7 用于监测条件的自动子程序

经常需要不用主机或DMC-2x00程序介入而连续监测某些条件，DMC-2x00能够在后台监测几个重要条件。这些条件有：检查限位开关、定义的输入、位置误差或命令误差等。在应用程序中插入特殊、预先定义的标号使自动监测功能生效。

6.7.1 预先定义的自动子程序标号有：

子 程 序	说 明
# LIMSWI	任意轴限位开关变低
# ININT	由II变低所规定的输入
# POSERR	由ER所指定的位置误差超限
# MCTIME	运动完成超时，超时周期由命令TW设定
# CMDERR	无效命令
# COMINT	通信中断子程序
# TCPERR	TCP/IP通信错误（只用于DMC2100/2200）

例如，当任意一个轴超过位置误差上限时，就会自动执行# POSERR子程序。# POSERR子程序中的命令译出哪个轴处于报警中并采取恰当行动。在另一个例子中，用# ININT标号来指向输入中断子程序。当指定的输入出现时，就会执行该程序。

注1：应用程序必须运行用于自动监测功能子程序。

注2：通信中断不适用于DMC-2100。

6.7.2 举例

举例1——限位开关

限位开关一出现，此程序就打印信息。注意，对于# LIMSWI子程序来说，DMC-2x00必须从执行着来自存储器的应用程序。这是一个非常简单的程序，不做任何事但只循环一个语句，例如# LOOP; JP# LOOP; EN。即使在执行“虚拟”应用程序当中，仍然可以从PC发送像JP5000那样的运动命令。

指 令	说 明
: ED	编辑方式
000# LOOP	虚拟程序
001 JP # LOOP; EN	跳转到循环程序

002 # LIMSWI	限位开关标号
003 MG "LIMIT OCCURRED"	打印信息
004 RE	返回主程序
<control> Q	退出编辑方式
: XQ # LOOP	执行虚拟程序
: JG 5000	Jog运动
: BGA	开始运动

现在，当A轴正向限位开关出现时，就会执行# LIMSWI。

几点注意事项：

- 1) 用RE命令从# LIMSWI子程序返回
- 2) 如果限位开关保持有效状态，就会再次执行# LIMSWI子程序
- 3) 当正在命令电机运动时，就只执行# LIMSWI子程序。

举例2——位置误差

指 令	说 明
: ED	编辑方式
000 # LOOP	虚拟程序
001 JP # LOOP; EN	循环
002 # POSERR	位置误差子程序
003 V1=_TEX	读位置误差
004 MG "EXCESS POSITION ERROR"	打印信息
005 MG 'ERROR=',V1=	打印误差
006 RE	从误差子程序返回
<control> Q	退出编辑方式
: XQ # LOOP	执行虚拟程序
: JG 100000	以高速进行Jog 运动
: BGX	开始运动

举例3——输入中断

指 令	说 明
# A	标号
II1	由输入点1产生输入中断
JG 30000, , , 60000	Jog运动
BG A D	开始运动
# LOOP; JP# LOOP; EN	循环
# ININT	输入中断
STXW; AM	停止运动
# TEST; JP # TEST, @IN[1]=0	若输入点1仍为低电平，循环测试
JG 3000, , , 6000	Jog运动
BG A D	开始运动
RI0	从中断程序返回到主程序，并不对条件启动重新使能

举例4——运动完成超时

指 令	说 明
# BEGIN	开始主程序
TW 1000	设置超时基准1000ms

PA 10000	绝对位移10000
BG A	开始运动
MC A	运动完成条件启动
EN	结束主程序
# MCTIME	运动完成子程序
MG "A fell short"	发送信息
EN	结束程序

如果A轴在运动结束后的1秒之内没有到达命令位置，此程序就会送出信息“A fell short”。

举例5——命令错误

指 令	说 明
# BEGIN	开始主程序
IN "ENTER SPEED", SPEED	提示输入速度
JG SPEED; BGA	开始运动
JP # BEGIN	重复循环
EN	结束主程序
# CMDERR	命令错误子程序
JP# DONE,_ED<>2	检查第2行是否出错
JP# DONE,_TC<>6	检查是否超出范围
MG 'SPEED TOO HIGH'	发送信息
MG "TRY AGAIN"	发送信息
ZS1	调整堆栈
JP # BEGIN	返回到主程序
# DONE	如果有其它错误，结束程序
ZS0	0堆栈
EN	结束程序

上述程序提示操作人员输入JOG速度值，如果操作人员输入的数值超出范围（大于8 x106），通过执行# CMDERR子程序，提示操作人员输入一个新速度值。

在多任务应用中，有另外的方法用于处理不同通道中的命令错误。下表所列的特殊操作数与XQ命令配合使用就能使控制器跳过或重新处理无效命令。

特殊操作数

操 作 数	说 明
_ED1	返回产生错误的通道号
_ED2	重试无效命令（操作数读取无效命令所处位置）
_ED3	跳过无效命令（操作数读取无效命令之后的命令位置）

以下列格式与XQ命令一起使用操作数：

XQ_ED2（或_ED3），_ED1，1

其中：在命令行末尾的‘，1’表示重新开始；因此，当执行上面的格式时，不会去掉已有的程序堆栈。

下例示出使用操作数的误差修正子程序。

举例6——命令错误（使用多任务时）

指 令	说 明
#A	开始通道0（连续循环）

JP# A	
EN	通道0结束
# B	开始通道1
n=-1	建立新变量
KPn	设置KP为N的值，此为无效值
TY	发布无效命令
EN	通道1结束
# CMDERR	开始命令错误子程序
IF _TC=6	如果错误，误差超出范围（KP-1）
N=1	设置N为有效数值
XQ _ED2,_ED1,1	重试KP N命令
ENDIF	
IF _TC=1	如果错误属无效命令(TY)
XQ _ED3,_ED1,1	跳过无效命令
ENDIF	
EN	命令错误子程序结束

举例7——通信中断

用DMC-2x10使A轴从0~10000往返运动4次，通过来自辅助串口终端的输入能使此运动暂停、重起和停止。

指 令	说 明
# BEGIN	程序标号
CC 9600,0,0,0	设置辅助串口通信配置
CI 2	设置辅助串口通信中断
MG {P2}"Type 0 to stop motion"	辅助串口信息输出
MG {P2}"Type 1 to pause motion"	辅助串口信息输出
MG {P2}"Type 2 to resume motion"	辅助串口信息输出
RATE=2000	速度变量
SPA=RATE	设置A轴运动速度
# LOOP	循环标号
PAA=10000	移动到绝对位置10000
BGA	开始A轴运动
AMA	等待运动完成
PAA=0	移动到绝对位置0
BGA	开始A轴运动
AMA	等待运动完成
JP # LOOP	继续循环进行往复运动
EN	结束主程序
# COMINT	中断子程序
JP # STOP, P2CH="0"	检查S (停止运动)
JP # PAUSE, P2CH="1"	检查P (暂停运动)
JP # RESUME, P2CH="2"	检查R (重起运动)
EN1, 1	
# STOP	停止运动子程序

STA; ZS; EN	停止A轴运动; 程序堆栈置0; 程序结束
# PAUSE	暂停运动子程序
RATE=_SPA	保存A轴运动当前速度值
SPA=0	将A轴速度设为0(允许暂停)
EN1,1	条件启动和通信中断重新使能
# RESUME	重起运动子程序
SPA=RATE	将A轴速度设为初始速度
EN1,1	条件启动和通信中断重新使能

举例8——Ethernet通信错误

在DMC-2100/2200中运行这个简单程序并指出（通过串口）通信端口何时失败。通过监控串口，若有必要，用户能够重新建立通信。

指 令	说 明
# LOOP	简单程序循环
JP# LOOP	
EN	
# TCPERR	Ethernet通信错误自动子程序
MG{P1}_IA4	将信息发送到串口，指明哪个端口没有接收正确的响应。
RE	

7. 数学及函数表达式

7.1 数学算子

为了数据变换，DMC-2x00具有如下数学运算功能

算 子	功 能
+	加
-	减
*	乘
/	除
&	逻辑与
	逻辑或（在有些计算机中，实竖线表现为虚线）
()	圆括号

加、减、乘运算的数值范围为±2147483647.9999,除运算的精确度为1/65000。

数学运算从左到右执行。圆括号内的计算优先进行。

SPEED=7.5*V1/2	变量SPEED等于V1乘以7.5再除以2
COUNT=COUNT+2	变量COUNT等于当前值加2.
RESULT=_TPA-(@COS[45]*40)	把A轴位置值减去28.28放入变量RESULT。 40 * COS 45° 为28.28
TEMP=@IN[1]&@IN[2]	只有当输入点1和输入点2均为高电平时,变量TEMP 等于 1

7.2 逐位操作算子

数学算子&和|是逐位操作算子。算子&是逻辑与，算子|是逻辑或。这些算子对任意有效的DMC-2x00 数字操作数（包括变量、数组元素、数字值、函数、键字和算术表达式）进行位操作

运算。位操作算子也可以与字符串一起使用。这对于把输入字符串拆分字符来说非常有用。采用输入命令进行字符串输入时，输入变量会拥有多达 6 个字符。这些字符合并成一个单值，表示为 32bit 整数和 16bit 小数。每个 ASCII 字符表达为 1 个字节（8bit），因此，输入变量就拥有 6 个字符。字符串的第一个字符置于变量的最高字节，而最后一个字符放置于小数点的最低有效字节。能通过下例所示的位操作将字符逐个分开。

指 令	说 明
# TEST	开始主程序
IN "ENTER", LEN{S6}	将6个字符的字符串放入变量“LEN”
FLEN=@FRAC[LEN]	将变量“FLEN”定义为变量“LEN”的小数部分
FLEN=\$10000*FLEN	左移FLEN 32 bits (IE – 将小数点部分FLEN转换成整数)
LEN1=(FLEN&\$00FF)	屏蔽FLEN最高字节并将此值设置成变量“LEN1”
LEN2=(FLEN&\$FF00)/\$100	让变量“LEN2” = FLEN的最高字节
LEN3=LEN&\$000000FF	让变量“LEN3” = LEN的最低字节
LEN4=(LEN&\$0000FF00)/\$100	让变量“LEN4” = LEN的第2字节
LEN5=(LEN&\$00FF0000)/\$10000	让变量“LEN5” = LEN的第3字节
LEN6=(LEN&\$FF000000)/\$1000000	让变量“LEN6” = LEN的第4字节
MG LEN6 {S4}	显示“LEN6”为4字符的字符串信息
MG LEN5 {S4}	显示“LEN5”为4字符的字符串信息
MG LEN4 {S4}	显示“LEN4”为4字符的字符串信息
MG LEN3 {S4}	显示“LEN3”为4字符的字符串信息
MG LEN2 {S4}	显示“LEN2”为4字符的字符串信息
MG LEN1 {S4}	显示“LEN1”为4字符的字符串信息
EN	

此程序将接受 6 个字符的字符串输入，解析各个字符，然后显示各字符。也要注意，用于屏蔽的值以十六进制表示（由引号符‘\$’标注），对于更多的相关信息，参见“发送信息”一节。

为了进一步说明，如果用户在输入提示处输入字符串“TESTME”，控制器就会响应如下：

T	来自命令MG LEN6 {S4}的响应
E	来自命令MG LEN5 {S4}的响应
S	来自命令MG LEN4 {S4}的响应
T	来自命令MG LEN3 {S4}的响应
M	来自命令MG LEN2 {S4}的响应
E	来自命令MG LEN1 {S4}的响应

7.3 函数

函 数	说 明
@SIN[n]	n的正弦 (n以度为单位，分辨率为1/64000度，最大±4 x 10 ⁹)
@COS[n]	n的余弦 (n以度为单位，分辨率为1/64000度，最大±4 x 10 ⁹)
@TAN[n]	n的正切 (n以度为单位，分辨率为1/64000度，最大±4 x 10 ⁹)
@ASIN[n]*	n的反正弦，在-90°和+90°之间，角度分辨率为1/64000度
@ACOS [n]*	n的反余弦，在0°~180°之间，角度分辨率为1/64000度
@ATAN [n]*	n的反正切，在-90°和+90°之间，角度分辨率为1/64000度
@COM[n]	n的2的补码
@ABS[n]	n的绝对值

@FRAC[n]	n的小数点部分
@INT[n]	n的整数部分
@RND[n]	n的求整（如果n的小数点部分 ≥ 0.5 ，则上进取整）
@SQR[n]	n的平方根（精确度为 ± 0.0001 ）
@IN[n]	返回通用输入点n的数字输入（其中n以1开始）
@OUT[n]	返回通用输出点n的数字输出（其中n以1开始）
@AN[n]	返回通用模拟输入n的模拟输入值（其中n以1开始）

*注：这些函数是多值函数，可以用应用程序找到正确数值范围。函数可以与数学表达式结合使用，数学表达式的执行次序是从左到右，且能够用圆括号相括而优先执行。

指 令	说 明
V1=@ABS[V7]	变量V1等于变量V7的绝对值.
V2=5*@SIN[POS]	变量V2等于5倍变量POS的正弦
V3=@IN[1]	变量V3等于输入点1的数字值
V4=2*(5+@AN[5])	变量V4等于模拟输入点5的值加5,然后乘以2

8. 变量

对于要求参数为变量的应用，DMC-2x00提供了254个变量，这些变量可以是数值或字符串。用户可以把某些参数（如位置或速度等）定义为变量，以变量形式写入程序；其后，再由操作人员分配变量值或者通过程序计算确定变量值。例如，定长裁断应用就需要切长为变量。

指 令	说 明
PR POSA	把变量POSA分配给PR命令
JG RPMB*70	把变量RPMB乘以70分配给JG命令

8. 1 可编程变量

DMC-2x00允许用户创建254个变量，各变量可以用8个字符构成的变量名加以定义。变量名必须以阿拉伯字符开头，不过，在变量名的其余部分允许用数字，但不允许有空格。变量可以是大小写，或者任意组合。变量属字盘敏感型；SPEEDC \neq (不等于) Speed C。变量名不应该与DMC-2x00指令相同。例如，PR就不适合用作变量名。

有效及无效变量名的例子有：

有效变量名：

POSA

POS1

SPEEDC

无效变量名

REALLONGNAME	；	不能多于8个字符
123	；	不能以数字开始
SPEED C	；	变量名中不能有空格

8. 2 给变量分配值

分配的值可以是数字、内部变量、键字、函数、控制参数和字符串；数字变量值的范围是4字节整数（ 2^{31} ）后跟2字节小数（ $\pm 2,147,483,647.9999$ ）。

8. 3 可以用等号把数值分配给可编程变量

可以用任意有效函数把值分配给变量。例如， $V1=@ABS[V2]$ 或 $V2=@IN[1]$ 。也允许有算术运算。

要分配字符串值，字符串必须在引号内。字符串变量可以含有6个字母，这些字母必须在引号中。

指 令	说 明
POSA=_TPA	把从TPA命令返回的值分配给变量POSA
SPEED=5.75	把5.75分配给变量SPEED
INPUT=@IN[2]	把输入点2的逻辑值分配给变量INPUT
$V2=V1+V3*V4$	将 $V1+V3*V4$ 的值分配给变量V2.
VAR="CAT"	将字符串CAT分配给VAR

8. 4 将变量值分配给控制器参数

可以将变量值分配给控制器参数（如GN或PR）

PR V1	将V1分配给PR命令
SP _VSS*2000	将_VSS*2000分配给SP命令

8. 5 在终端上显示变量值

可以用格式，变量=将变量发送到显示屏。例如， $V1=$ ，返回变量V1的值。

举例1、将变量用于操纵棒（Joystick）

下例读取A-B轴操纵棒的电压值，并将它分配给变量VA和VB，以比例速度驱动电机运动，其中：

10 V= 3000 rpm = 200000 cts/sec	
速度/模拟输入（比例速度）= 200000/10 = 20000	
指 令	说 明
# JOYSTIK	标号
JG 0,0	设定JOG方式
BGAB	开始运动
# LOOP	循环
$VA@AN[1]*20000$	读取操纵棒A
$VB@AN[2]*20000$	读取操纵棒B
JG VA,VB	以变量VA,VB进行JOG运动
JP# LOOP	重复循环
EN	结束

9. 操作数

操作数允许把DMC控制器运动或状态参数并入可编程变量和表达式。大多数DMC-2x00命令都有一个等效的操作数，通过在DMC-2x00命令之前添加下划线即指定为操作数。本书第二篇命令手册中明确了哪些命令有相应的操作数。

状态命令，如TP读回实际位置值，而作用命令，如KP或SP就读回DMC寄存器中的值；跟随命令之后需要轴指定。

指 令	说 明
POSA=_TPA	将来自A轴实际位置寄存器的值分配给变量POSA.
JP # LOOP, _TEA>5	如果A轴位置误差寄存器中的值大于5，就跳转到#LOOP

JP # ERROR, _TC=1

如果错误代码等于1，就跳转到#ERROR

可以在表达式中使用操作数，并将其分配给可编程变量，但不能对其分配数值，例如：_TPA=2是无效的。

特殊操作数（键字）

DMC-2x00提供了一些附加操作数，用于存取那些不能用标准DMC-2x00命令存取的内部变量。

键 字	说 明
_BGn	*如果‘n’轴运动完成，返回值为1，否则为0
_BN	*读取印刷电路板的序号
_DA	*读取数组变量数
_DL	*读取用于编程的可用标号数
_DM	*读回可用的数组存储区
_HMn	*读回原点开关的状态（0或1）
_LFn	读回‘n’轴正向限位开关输入状态（0或1）
_LRn	读回‘n’轴负向限位开关输入状态（0或1）
_UL	读回可用变量数
TIME	空运行实时时钟（减少2.4%，——由上电进行复位）注意：TIME不使用下划线（-）作为其它键字

带有‘*’号的键字有相对应的命令，而键字_LF、_LR和TIME没有相应的命令。

V1=_LFA 将A轴正向限位开关状态分配给V1

V3=TIME 将时钟当前值分配给V3

V4=_HMD 将D轴原点开关输入逻辑状态分配给V4

10. 数组

为了存储、采集数据，DMC-2x00提供了8000个元素用的数组空间。数组是一维的，可以定义30个不同数组。各个数组元素都有一个4字节整数（ 2^{31} ），后跟2字节小数的数值范围（±2147483647.9999），可以用数组来捕获实时数据，如位置、转矩和模拟输入值。在轮廓方式下，在记录和录返应用中，可以很方便地用数组来保存位置轨迹的各点坐标值。

10. 1 定义数组

用命令DM定义数组，用户必须指定要放入的数组名和入口号。数组名可以含有8个字符，以大写阿拉伯字符开头，再定义数组中的入口号，入口号以中括号[]结束。

DM POSA[7] 定义带有7个入口的数组名POSA

DM SPEED[100] 定义带有100个入口的数组名SPEED

DM POSA[0] 空数组空间

10. 2 数组入口分配

像变量一样，可以给各数组元素分配一个值。分配的值可以是数组或从指令、函数、键字来的返回值。

数组元素地址分配从count 0 开始，例如，POS A数组中的第1个元素（用DM POSA[7]定义）就指定为DM POSA[0]。

用等号将数组分配给数组入口，通过指定带有相应数组名的元素号，每次一个元素，逐个分配。

注意：在分配入口值之前，必须用命令DM定义数组。

DM SPEED[10] 定义数组Speed大小

SPEED[1]=7650.2	给数组的第1个元素分配值7650.2
SPEED[1]=	读回数组第一个元素值
POSLA[10]=_TPA	给第10个元素分配A轴实际位置值
CON[2]=@COS[POS]*2	给第2个数组元素分配POS 乘以2的余弦值
TIMER[1]=TIME	给第1个数组元素分配TIME

10. 3 用变量给数组元素分配地址

数组元素号也可以是一个变量。这样，就可以用一个计数器按序给数组入口分配值。

指 令	说 明
#A	程序标号
COUNT=0; DM POS[10]	初始化计数器并定义数组
# LOOP	开始循环
WT 10	等待10 msec
POS[COUNT]=_TPA	将位置值记录入数组元素
POS[COUNT]=	报告位置
COUNT=COUNT+1	计数器加1
JP # LOOP,COUNT<10	循环，直到保存完10个元素
EN	结束

上例表示，以每10ms一个值的速率将10个位置值存入10个数组元素，这10个值存入数组名POS中，变量count用来使数组元素计数器增1。上例也可用下述的自动数据捕获功能来实现。

10. 4 上载、下载数组到板上存储器

也可以用QU和QD命令上载和下载数组。

QU array[], start, end, delim

QD array[], start, end

其中：array是数组名，例如：A[]

start是第一个数组元素（缺省值=0）

end是最后一个数组元素（缺省值=最后一个元素）

delim定义数组数据用逗号“，”（delim=1）还是用回车（delim=0）来分开。

用<control>Z，<control>Q，<control>D 或 \来终止文件。

10. 5 将数据自动捕获到数组中

DMC-2x00控制器有一个用于数据自动捕获的特殊功能，可以用于对诸如位置、位置误差、输入、转矩等进行自动捕获，并存入数组。这对于示教运动轨迹或观察系统性能而言非常有用。可以捕获4种数据，并存入4个数组。可以指定捕获速率或时间间隔，可以按一次事件或一个循环连续记录进行记录。

10. 5. 1 命令汇总——自动数据捕获

命 令	说 明
RA n[], m[], o[], p[]	为数据捕获选择4个数组，必须用DM命令定义数组
RD type1, type2, type3, type4	选择要记录的数据类型，其中：type1, type2, type3, type4代表各种数据类型（见下表），数据类型的次序很重要，与RA命令中n, m, o, p数组的次序相对应。
RC n,m	RC命令开始数据采集，设置数据捕获时间间隔，其中：n是

	1~8的整数，指定数据之间相隔 2^n ms；m是选择项，规定要被捕获的元素数量；若不定义m，元素数量就缺省为由DM命令定义的最小数组。当m是负数时，就以循环方式连续进行记录。_RD是记录指针，指示下一个数组元素的地址。n=0则停止记录。
RC?	返回值为0或1，其中：0表明不记录，1表示记录在进行。

10. 5. 2 用于记录的数据类型

数据类型	说明
_DEA	第2编码器反馈位置（双编码器使用时）
_TPA	编码器反馈位置
_TEA	位置误差
_SHA	命令位置值
_RLA	锁存的位置值
_TI	输入点
_OP	输出点
_TSA	开关点（只有bit 0-4有效）
_SCA	停止代码
_NOA	状态位
_TTA	转矩值（报告数字值 ± 32544 ）
_AFA	模拟输入值（字母对应输入，例如，AFA=第1个模拟输入，AFB=第2个模拟输入）

注意：表中A可以用B、C、D、E、F、G、H取代，用于捕获其它轴的数据。

10. 5. 3 操作数汇总——自动数据捕获

操作数	说明
_RC	返回值为0或1，其中：0表示不记录，1表示记录在进行
_RD	返回值为下一个数组元素的地址

10. 5. 4 举例——记录入数组

指令	说明
# RECORD	标号
DM APOS[300], BPOS[300]	定义A、B轴位置数组
DM AERR[300], BERR[300]	定义A、B轴误差数组
RA APOS[], AERR[], BPOS[], BERR[]	为捕获选择数组
RD _TPA, _TEA, _TPB, _TEB	选择数据类型
PR D000,20000	指定运动距离
RC1	以2 msec 速率开始记录
BG AB	开始运动
#A; JP # A, RC=1	循环
MG "DONE"	发送信息
EN	结束程序
# PLAY	录返程序标号
N=0	初始化计数器
JP# DONE, N>300	如果完成，退出

N=	打印计数器值
A POS[N]=	打印A轴位置
B POS[N]=	打印B轴位置
AERR[N]=	打印A轴误差
BERR[N]=	打印B轴误差
N=N+1	计数器加1
#DONE	标号
EN	结束程序

10. 5. 5 取消数组空间分配

用DA命令后跟数组名就可以取消数组空间分配，DA*[0]将所有数组分配取消。

11. 数据输入（数字和字符串）

11. 1 数据输入

用命令IN来提示用户输入数字或字符串，使用IN命令，用户可以把信息放在双引号中，指定信息提示内容。控制器执行IN命令时，就会等待输入数据。把输入分配给所指定的变量或数组元素。

举例1——数据输入

```
#A
IN "Enter Length", LenA
EN
```

在此例中，信息“Enter Length”显示在计算机屏幕上，控制器等待操作人员输入一个长度值，操作人员就输入已分配给变量LenA的值。（注意：在输入信息末尾的逗号和变量名之间不能有空格）。

举例2——定长裁断

在此例中，材料的长度要事先规定好，当运动完成时，使刀头工作，切断材料。长度值是一个变量，提示操作者以英寸输入。切割运动由连接到输入点1的启动按钮启动。负载与螺距为2吋的丝杠相连接，2000cts/rev编码器安装在电机上，分辨率为4000cts/inch。下面的程序对长度用变量LEN，用IN命令来提示操作人员输入长度值，然后输入的长度值就分配给变量LEN。

指 令	说 明
# BEGIN	标号
AC 800000	加速度
DC 800000	减速度
SP 5000	速度
LEN=3.4	初始长度
#CUT	Cut子程序
AI1	等待启动信号
IN "enter Length(IN)", LEN	提示操作人员以英寸为单位输入长度值
PR LEN *4000	以cts指定位置值
BGX	开始运动移动材料
AMX	等待运动完成
SB1	对切刀设置输出，使切刀动作
WT100; CB1	等待100 msec，然后关断切刀
JP # CUT	重复切割子程序

11. 2 操作器数据输入方式

操作器数据输入方式通过辅助RS-232口提供了非缓冲的数据输入方法，在这种方式下，DMC-2x00具有接收字符的缓存器。这种方式也只可用于执行应用程序时。

操作器数据输入方式只可以规定为port 2，用\或<escape>键退出该方式。

注意：操作器数据输入方式不能用于高速数据传输。

将CC命令的第3区设置为0，就设置成了操作器数据输入方式。

要获取或解译操作器数据方式中的字符，DMC-2x00提供了如下一些特殊键字：

键 字	功 能
P2CH	读取接收到的最后一个字符
P2ST	读取接收到的字符串
P2NM	读取接收到的数字
P2CD	读取状态码： -1 方式无效 0 未收到任何信息 1 收到字符，但没有<enter> 2 收到字符串，但没有数字 3 收到数字

注意：在读取对应的字符串或数字之后，P2CD的值返回到0。这些键字可以用在应用程序中对数据进行译码，也可以用在带有逻辑算子的条件语句中。

举例：

指 令	说 明
JP # LOOP, P2CD<>3	检查状态码是否为3 (接收到的数字)
JP # P, P2CH="V"	检查收到的最后一个字符是不是V
PR P2NM	将接收到的数分配给位置
JS # XAXIS, P2ST="X"	检查收到的字符串是不是X

11. 3 使用通信中断（只用于DMC-2000）

DMC-2x00对通信提供了特殊中断功能，使用户通过输入点来中断应用程序；用CI命令使中断使能，此命令的句法是CIn：

其中：n = 0	不中断Port 2
n = 1	用<enter> 中断Port 2
n = 2	用任意字符中断Port 2
n = -1	清除缓存器中的所有字符

COMINT标号用于通信中断。例如，能将DMC-2x00配置成由Port2接收到的任意字符来中断。当收到字符时，就进入# COMINT子程序，子程序对字符进行译码。在子程序的末尾，使用EN命令。EN，1会使中断重新使能，并返回到调用中断的程序行；EN只返回到调用中断的程序行，不对中断重新使能。带有任何自动子程序的程序在任何时刻，必须只能在0通道运行。

举例：

用DMC-2x00使A、B轴进行JOG运动，此程序一上电就自动开始运行，用户可以从主串口终端输入数值。在运行期间，通过指定轴字母后跟新速度值，可以更改运行速度；S使两个轴运动停止。

指 令	说 明
# AUTO	标号（自动运行程序）
SPEEDA=10000	置A轴初始速度
SPEEDB=10000	置B轴初始速度
CI 2	设置Port 2为字符中断方式
JG SPEEDA, SPEEDB	指定A、B轴为JOG速度
BGAB	开始运动
# PRINT	发送信息到终端的子程序
MG{P2}"TO CHANGE SPEEDS"	发送信息
MG{P2}"TYPE A OR B"	
MG{P2}"TYPE S TO STOP"	
# JOGLOOP	循环改变JOG速度
JG SPEEDA, SPEEDB	设置新JOG速度
JP # JOGLOOP	
EN	结束主程序
# COMINT	中断子程序
JP # A, P2CH="A"	检查A
JP # B, P2CH="B"	检查B
JP # C, P2CH="S"	检查S
ZS1; CI2; JP# JOGLOOP	如果没有A, B, S, 则跳转
#A; JS# NUM	
SPEEDA=VAL	新A速度
ZS1; CI2; JP# PRINT	跳转到# PENT子程序
#B; JS# NUM	
SPEEDB=VAL	新B速度
ZS1; CI2; JP# PRINT	跳转到# PENT子程序
#C; ST; AMX; CI-1	停止S运动
MG{^8}, "THE END"	
ZS; EN, 1	结束 中断重新使能
# NUM	输入新JOG速度的子程序
MG "ENTER",P2CH{S},"AXIS SPEED"	提示输入速度值
{N}	
# NUMLOOP; CI-1	检查输入
# NMLP	检查从终端输入的子程序
JP # NMLP, P2CD<2	若为字符串, 跳入出错子程序
JP # ERROR, P2CD=2	读值
VAL=P2NM	
EN	结束子程序
# ERROR; CI-1	错误检查子程序
MG "INVALID-TRY AGAIN"	提示信息
JP # NMLP	
EN	结束

11. 4 输入字符串变量

可以用指定器{Sn}输入带有6个字符的字符串变量, 其中, n代表要输入的字符串数量。如果

没有指定n, 就将接受6个字符。例如, IN "Enter A,B or C", V{S}表示要输入的字符串变量。

DMC-2x00对所有变量按6字节信息存储, 当把变量指定为数值时, 变量值就表示为4字节整数和2字节小数; 当把变量指定为字符串时, 变量可拥有6个字符(每个ASCII字符1字节)。用IN命令进行字符串输入时, 第1个输入字符放在变量的最高字节, 而最后一个字符放在小数的最低有效字节。可以用位操作将字符逐个分开。

12. 数据输出(数字和字符串)

可以用几种方法从控制器输出数字和字符串数据。信息命令MG能输出字符串和数字数据。还有, 能用查询命令对控制器进行命令, 以返回变量和数组值以及其它信息。详见第5章所描述的查询命令。

12. 1 发送信息

用信息命令MG可以将信息发送到总线, 此命令将指定的文本、数字或字符串数据从变量或数组发送到屏幕。

用双引号指定文本字符串, 而用变量或数组名指定变量或数组数据。例如:

```
MG "The Final Value is", RESULT
```

除变量、函数和命令之外, 在信息命令中还可以使用响应。例如:

```
MG "Analog input is", @AN[1]
```

```
MG "The Position of A is", _TPA
```

12. 1. 1 指定信息发送口

通过缺省设置, 将信息通过由USB/Ethernet指拨开关所指定的通信口进行发送, 这些指拨开关一上电的状态就会决定是通过USB口(DMC-2000)还是Ethernet(DMC-2100/2200)或者主串口发送信息。不过, 可以用指定器对口进行指定, {P1}对应主串口, {P2}对应辅串口, {U}对应USB口, {E}对应Ethernet口。

```
MG {P2} "Hello World"          发送信息到辅串口
```

12. 1. 2 格式化信息

用指定器{Sn}对字符串变量进行格式化, 其中, n是字符数, 1~6。例如:

```
MG STR {S3}
```

这条语句返回字符串变量名为STR的3个字符。用{Fn.m}表达式后跟完整的MG语句就可以对数字数据进行格式化。{Sn.m}以十六进制而不是十进制对数据进行格式化, 将实际数值格式化成小数点左面n个字符, 小数点右面m个字符。有效值前面的0用来显示指定的格式。

例如:

```
MG "The Final Value is", RESULT {F5.2}
```

如果变量RESULT的值等于4.1, 上述语句返回如下:

```
The Final Value is 00004.10
```

如果变量RESULT的值等于999999.999, 上述语句返回如下:

```
The Final Value is 99999.99
```

信息命令通常发送跟着语句的回车和行进。在语句末尾发送{N}就可以阻止回车和行进。当文本字符串需要处于数值前后时, 这就很有用。

例如:

```
#A
```

```
JG 50000; BGA; ASA
```

```
MG "The Speed is", _TVA {F5.1} {N}
```

```
MG "counts/sec"
```

EN

当执行上述程序时，屏幕上就会显示出：

The speed is 50000 counts/sec

12. 1. 3 用MG命令配置终端

能用MG命令对终端进行配置。可以用格式{^n}发送任意ASCII字符，

其中：n是1~255之间的任意整数。

例如：

MG {^07} {^255}

将用7和255表示的ASCII字符发送到总线。

12. 1. 4 信息函数汇总

函 数	说 明
“ ”	围绕文本字符串
{Fn.m}	格式化数值，小数点左面n位十进制数，右面m位
{P1}, {P2}, {U} 或 {E}	发送信息到主串口、辅串口、USB或Ethernet口
{ \$n.m}	以十六进制对数值格式化
{^n}	发送由整数n所规定的ASCII字符
{N}	阻止回车/行进
{Sn}	发送字符串变量前面n个字符，其中n=1~6

12. 2 显示变量和数组

可以用格式，变量= 或 数组[x]=. 将变量或数组发送到屏幕上。例如，

V1=, 就返回V1的值。

举例——显示变量和数组元素

指 令	说 明
# DISPLAY	标号
DM POSA[7]	定义数组POSA为7个入口
PR 1000	位置命令
BGX	开始运动
AMX	在运动完成后
V1=_TPA	分配变量V1
POSA[1]=_TPA	分配第1个入口
V1=	显示V1

12. 3 查询命令

DMC-2x00有一组直接查询控制器的命令。当输入这些命令时，就以十进制返回所要的数据，通过回车和行进而进入下一行。返回的数据格式能用位置格式化命令（PF）和前位充0命令（LZ）进行修改。关于查询命令的完整内容参见第5章。

12. 3. 1 用PF命令对查询命令所产生的回应进行格式化

PF命令能修改由查询命令所返回值的格式：

BL ?	LE ?
DE ?	PA ?

DP ?	PR ?
EM ?	TN ?
FL ?	VE ?
IP ?	TE
TP	

用PF命令可以对十进制或十六进制数值格式化，使其该数的小数点右面和左面具有所规定的位数。命令格式如下：

PF m.n

其中：m是小数点左面的位数（0~10），n是小数点右面的位数（0~4），负的m则指定为十六进制格式。

返回十六进制值由\$开头，以2的补码表示。十六进制值应该按带符号2的补码进行输入，其中，负数有负号。缺省格式为PF10.0。如果由PF所规定的十进制位数小于实际值，那么9就出现在所有的十进制位。

举例：

指 令	说 明
: DP21	定义位置
: TPA	报告位置
0000000021	缺省格式
: PF4	修改格式为4位
: TPA	报告位置
0021	新格式
: PF-4	修改成十六进制格式
: TPA	报告位置
\$0015	十六进制值
: PF2	修改格式为2位
: TPA	报告位置
99	如果位置值大于99，就返回99

12. 3. 2 从查询命令响应中去掉前面的0

可以用命令LZ去掉由查询命令所返回值前面的0。

指 令	说 明
LZ0	使LZ功能无效
TP	报告位置
-0000000009, 0000000005	返回值(带有前0)
LZ1	使LZ功能使能
TP	报告位置
-9, 5	返回值 (无前0)

12. 3. 3 查询命令返回值的局部格式化

可以对查询命令返回值进行局部格式化。要想进行局部格式化，在查询命令的同一行使用命令{Fn.m}或{\$n.m}。符号F指出应该以十进制格式返回响应，而\$ 指定以十六进制返回。n是小数点左面的位数，m是小数点右面的位数。

指 令	说 明
TP {F2.2}	以十进制格式2.2报告位置
-05.00, 05.00, 00.00, 07.00	查询所得结果



以十六进制格式4.2报告位置

查询所得结果

12. 4. 1 用变量格式化 (VF) 命令对变量和数组元素进行格式化。VF命令格式如下:

十六进制返回值前缀\$, 以2的补码表示。

109

位输入数值，这就需要程序把输入数转换成计数单位，这可用计数值/rev与输入值相乘。

指 令	说 明
# RUN	标号
IN “ENTER # OF REVOLUTIONS”, N1	提示输入转速
PR N1*2000	转换成计数单位cts
IN “ENTER SPEED IN RPM”, S1	提示输入RPM值
SP S1*2000/60	转换成cts/sec
IN “ENTER ACCEL IN RAD/SEC2”, A1	提示输入加速度
AC A1*2000/(2*3.14)	转换成cts/sec ²
BG	开始运动
EN	结束程序

13. 硬件I/O

13. 1 数字输出

DMC-2x00有8点通用输出口和64点扩展I/O用于控制外部事件。用CO命令可以将扩展I/O配置成输入或输出，DMC-2x50~DMC-2x80还有8个附加输出点。用软件命令SB（置位）、CB（位清除）或OB（输出位）对输出口的各位进行置位或清除操作。

举例1——置位和清除位

指 令	说 明
SB6	将输出口6置于高电平
CB4	将输出口4清除为低电平

举例2——输出位

输出位（OB）命令常根据变量、数组、输入或表达式的值用于设置或清除输出。任何非0值都会使输出置位。

指 令	说 明
OB1, POS	如果变量POS不为0，则对输出点1置位，如果POS等于0，则清除输出点1。
OB 2, @IN [1]	如果输入点1为高电平，则对输出点2置位，如果输入点1为低电平，则清除输出点2
OB 3, @IN [1]&@IN [2]	如果输入点1和输入点2均为高电平，则对输出点3置位
OB 4, COUNT [1]	如果数组COUNT中的元素1不为0，则对输出点4置位

通过用指令OP（输出口）指定8位字来设置输出口。这条指令使用户能用单条命令对整个8位输出口的状态进行定义，其中，2⁰ 是输出点1, 2¹是输出点2，以此类推。1 表示输出点接通。

常常用输出来设置运动进行期间各继电器状态或外部开关和事件

举例3——在运动之后接通输出

指 令	说 明
# OUTPUT	标号
PR 2000	位置命令
BG	开始
AM	运动之后
SB1	输出点1置高电平
WT 1000	等待1000 msec
CB1	输出点1置低
EN	结束

13. 2 数字输入

用@IN[n]函数或TI命令可以对通用数字输入点状态进行存取。@IN[n]函数返回所指定输入点n的逻辑电平，其中，n是1~96。

举例1——用输入点来控制程序流程

指 令	说 明
JP # A,@IN[1]=0	如果输入点1为低，则跳转到# A
JP # B,@IN[2]=1	如果输入点2为高，则跳转到# B
AI 7	等待直到输入点7为高
AI -6	等待直到输入点6为低

举例2——由开关按钮启动运动

当用户将面板开关按为ON时，A轴电机必须以4000 cts/sec速度运转，当面板开关转换到OFF时，A轴电机必须停止运转。

方案：将面板开关接到DMC-2x00的输入点1，输入点1为高电平意味着开关处于ON位置。

指 令	说 明
# S; JG 4000	设置速度
AI 1; BGX	在输入点1变高后开始运动
AI -1; STX	在输入点1变低后停止运动
AMX; JP # S	运动完成后，重复循环
EN	

13. 3 辅助编码器输入

辅助编码器输入口可以用作为通用输入，对于各轴来说，控制器都有一个辅助编码器接口，每个辅助编码器口由两个输入CHA和CHB组成。辅助编码器输入地址分配为输入点81~96。

来自于辅助编码器的各路输入接口是差分线接收器，能够承受±12V之间的电压。输入已被配置成接受TTL电平信号。要连接TTL电平信号，只需简单地把信号连接到正输入端，各输入端不要连接。对于其它信号电平，应该把负输入端连接到1/2全程电压范围（例如，如果信号是0~12V逻辑，那么就把负输入端连到6V）。

例如：

DMC-2x10有一个辅助编码器接口，此接口有两个输入（CHA和CHB）。给CHA输入分配输入点81，CHB输入分配输入点82。要将此输入用作为2个TTL信号，第一个信号连接到AA+，第二个信号连接到AB+，AA-、AB-不要连接。要存取这个输入，用函数@IN[81] 和 @IN[82].即可。

注意：不能将辅助编码器用于配置为步进电机方式的任何轴。

13. 4 输入中断功能

DMC-2x00提供有输入中断功能，此功能会使程序自动执行跟在#ININT标号之后的指令。用II m, n, o命令使此功能使能，m代表起始输入，n代表范围内的最后输入，参数o是中断屏蔽。如果不用m和n，o就表示屏蔽号。1表示对中断进行使能的输入，其中， 2^0 是 bit 1, 2^1 是 bit 2，以此类推。例如，II, , 5 使输入点1和3 ($2^0 + 2^2 = 5$)使能。

任何一个指定输入点的低电平输入都会使#ININT子程序自动执行。用中断返回命令（RI）使程序运行从这个子程序返回到中断发生的地方。如果在执行#ININT子程序之后，要想返回到程序中的其它地方，就用零堆栈（ZS）命令后面跟着无条件跳转语句。

重要提示：要从# ININT 子程序返回，使用 RI 命令（而不是 EN）

举例——输入中断

指 令	说 明
# A	标号#A
II 1	由输入点1产生中断
JG 30000,-20000	设定A、B轴速度
BG AB	开始A、B轴运动
# B	标号# B
TP AB	报告A、B轴位置
WT 1000	等待1000 ms
JP # B	跳转到#B
EN	程序结束
# ININT	中断子程序
MG “Interrupt has occurred”	显示信息
ST AB	停止A、B轴运动
# LOOP; JP # LOOP, @IN[1]=0	循环直至中断清除
JG 15000, 10000	指定新速度
WT 300	等待300 ms
BG AB	开始A、B轴运动
RI	中断返回

13. 5 模拟输入

DMC-2x00提供8路模拟量输入。用命令@AN[n]函数可以读取这些点的电压值，其中：n代表模拟输入点1~8。AD转换的分辨率是12bit（16bit ADC为选择功能）。用模拟输入来读取特殊传感器的值，如温度、张力或压力等。

下例示出使电机跟随模拟信号运动的程序，第1个例子是点——点（PTP）运动，第2个例子表示连续运动。

举例1——位置跟随器（点——点）

目的：电机必须跟随模拟信号运动，当模拟信号变化到10V时，电机必须移动10000cts。

方法：读取模拟输入值，并命令A轴移动到10000cts处。

指 令	说 明
# POINTS	标号
SP 7000	速度
AC 80000; DC 80000	加、减速度
# LOOP	
VP=@AN[1]*1000	读入模拟输入值并计算位置
PA VP	命令位置
BGA	开始运动
AMA	运动完成后
JP # LOOP	重复循环
EN	结束

举例2——位置跟随器（连续运动）

方法：读入模拟输入值，计算命令位置值和位置误差。命令电机以与位置误差成比例的速度运转。

指 令	说 明
# CONT	标号
AC 80000; DC 80000	加减速速度
JG 0	开始JOG方式
BGX	开始运动
# LOOP	
VP=@AN[1]*1000	计算需要的位置
VE=VP-TPX	求出位置误差
VEL=VE*20	计算速度
JG VEL	改变速度
JP # LOOP	重复循环
EN	结束

14. DMC-2x00控制器的扩展I/O

DMC-2x00控制器有64个扩展I/O点，可以通过软件将这些I/O点以8bit增量配置成输入或输出。I/O点通过1个80pin高密度连接器与外部相连接。

14.1 配置DMC-2x00的I/O点

能将DMC-2x00系列控制器的64个扩展I/O点以8个块进行配置。扩展I/O表示为块2~9或bits17~80。

用命令CO将扩展I/O配置成输入或输出。CO命令有一个域：

CO n

其中，n是代表二进制数的十进制值。各位二进制数代表扩展I/O的一个块。当设置为1时，对应的块就配置为输出。

最低有效位代表块2，而最高有效位代表块9。可以用下式求出十进制值： $n = n_2 + 2*n_3 + 4*n_4 + 8*n_5 + 16*n_6 + 32*n_7 + 64*n_8 + 128*n_9$ ，其中， n_x 代表块。如果 n_x 值是1，那么8个I/O点的块就要配置成输出。若 n_x 的值为0，那么，8个I/O点的块就会配置为输入。例如，如果要把块4和块5配置成输出，就写CO 12。

8位一组I/O块	块	二进制表达	块的十进制值
17~24	2	2^0	1
25~32	3	2^1	2
33~40	4	2^2	4
41~48	5	2^3	8
49~56	6	2^4	16
57~64	7	2^5	32
65~72	8	2^6	64
73~80	9	2^7	128

确定n的最简单方法：

第1步：确定哪8位组I/O块要配置成输出

第2步：从表中，确定要配置为输出的各I/O块的十进制值

第3步：将第2步确定的所有值相加，即为用于n的值

例如，如果要将块2和块3配置为输出，那么n=3，并命令CO3。

注意：这个计算结果与公式等同， $n = n_2 + 2*n_3 + 4*n_4 + 8*n_5 + 16*n_6 + 32*n_7 + 64*n_8 + 128*n_9$ 其中， n_x 代表块。

14. 2 将输出状态存入非易失性存储器中

扩展I/O的配置和输出点的状态都能用BN命令存入EEPROM中。如果未做任何设置，就采用缺省值CO0（所有块均为输入）。

14. 3 存取扩展I/O

当配置为输出时，可以用SBn和CBn 命令对各个I/O点进行定义。(其中，n=1~8 和17~80)。。也可以用条件命令OBn 定义输出(其中 n=1~8和17~80)。

也可以用命令OP来设置输出位，指定为数据块。OP命令接纳5个参数。第1个参数设置控制器主出口的值(输出位1-8, 块0)。其余4个参数设置外部扩展I/O值：

OP m, a, b, c, d

其中m是代表bits 1-8的十进制值（0~255），而a, b, c, d代表扩展I/O，以16位连续组为单位，（其值从0~65535）。对于配置成为输入点的I/O点来说，忽略所给的参数，下表说明了用来配置输出点状态的参数。

参 数	块	位	说 明
M	0	1-8	通用输出
a	2, 3	17-32	扩展I/O
b	4, 5	33-48	扩展I/O
c	6, 7	49-64	扩展I/O
d	8, 9	65-80	扩展I/O

例如，如果将块8配置为输出，就可以用如下命令：

OP 7, , , , 7

此命令会将位1, 2, 3 (块0)和位65, 66, 67 (块8) 置为1。位4~8和位68~80置为0。其它所有位不受影响。

当存取配置为输入的I/O块时，使用TIn命令。参数‘n’ 对应要读取的块(n=0, 2, 3, 4, 5, 6, 7, 8, 9)。读回的值是相对应位的十进制代表值，能用@IN[n]函数查询各位。(其中，n=1~8或17~80)。如果输入以下命令：

MG @IN[17]

控制器就会返回块2的最低有效位的状态（假设块2设置为输入）。

14. 4 与Grayhill 或 OPTO-22 G4PB24接口

DMC-2x00控制器使用1个80 Pin高密度连接器，需要1根带有80pin高密度连接器的高密度连接电缆(CABLE-80)。把这根电缆转换成2个50 pin IDC连接器，此连接器与I/O安装导轨相兼容，如Grayhill 70GRCM32-HL和OPTO-22 G4PB24等。要转换80 pin电缆，使用CB-50-80适配器。连接到CB-50-80的50 pin扁平电缆直接连接到I/O安装导轨上。

当使用OPTO-22 G4PB24安装导轨时，用户只可以读取控制器上的48~64I/O点。块5和块9必须配置为输入，且通过I/O盒连接。

15. 应用举例

15. 1 线缆裁切器

操作人员按下启动开关，使电机前进10英寸线距，当运动停止时，控制器产生输出信号，启动切刀。切割完成周期为100 ms。

假设电机通过一个直径为2英寸的滚子驱动线缆，同时假定编码器分辨率为1000线/rev。由于

滚轮的圆周等于2 π 英寸，它对应于4000正交值，因此，1英寸行程就等于：

$$4000/2 \pi = 637 \text{ cts/inch}$$

这就是说，10英吋的距离等于6370 cts，如旋转速度为5 inches/sec，即等于3185 cts/sec。

输入信号可以用输入点1，而输出信号选为输出点1，电机速度轮廓和相关输入、输出信号示于图7.1。

程序按我们在程序#A中所定义的状态起动，这里，控制器等待输入点1的输入脉冲，一旦脉冲给出，控制器就开始正向运动。

正向运动一完成，控制器就输出20 ms的脉冲，然后在返回#A进行下一个新循环之前等待另外80ms。

指 令	说 明
#A	标号
AI1	等待输入点1
PR 6370	距离
SP 3185	速度
BGA	开始运动
AMA	运动完成后
SB1	输出点1置位
WT 20	等待20 ms
CB1	输出点1清0
WT 80	等待80 ms
JP # A	重复循环

开始脉冲 I1

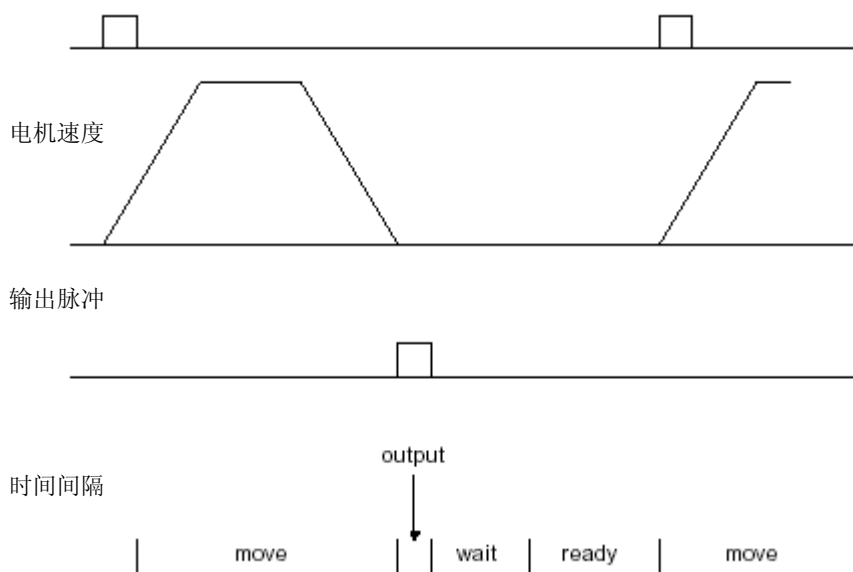


图7.1 – 电机速度和相关输入/输出信号

15. 2 X-Y工作台控制器（3轴应用）

一台X-Y-Z三轴系统必须切割出图7.2所示的形状，X-Y工作台做平面运动，而Z轴使切割刀具做上、下运动。

图7.2中的实线表示刀具行走的切割路径，要求进给速率为1 inch/sec；虚线表示刀具的非切割路径，运动速率为5inch/sec，加速度为0.1 g。

运动从A点开始，Z轴抬起，X-Y轴移动到B点的同时，Z轴向下运动，开始沿圆弧进行切割。圆弧加工运动一完成，Z轴上升抬起刀具，X、Y轴继续运动到C点，以此类推。

假定，3个轴的丝杆螺距都为0.1inch/rev，且编码器的分辨率为1000线/rev，由此得出：

$$1 \text{ inch} = 40,000 \text{ cts}$$

$$\text{速度为: } 1 \text{ in/sec} = 40,000 \text{ cts/sec}$$

$$5 \text{ in/sec} = 200,000 \text{ cts/sec}$$

加速度0.1g 等于：

$$0.1g = 38.6 \text{ in/sec}^2 = 1,544,000 \text{ cts/sec}^2$$

注意：圆弧半径为2" 或 80000 cts，加工起始点角度为270°，以CW（反方向）旋转360°，这样，就用如下指令规定轨迹：

CR 80000, 270, -360

进一步假定，Z轴移动2"，移动速度为2"/sec。用以下指令完成所要求的运动：

指 令	说 明
#A	标号
VM XY	X、Y轴圆弧插补
VP 160000,160000	矢量位置
VE	结束矢量运动
VS 200000	矢量速度
VA 1544000	矢量加速度
BG S	开始运动
AM S	运动完成时
PR, , -80000	Z轴下降

SP, , 80000	Z轴速度
BG Z	开始Z轴运动
AM Z	等待Z轴运动完成
CR 80000, 270, -360	圆弧
VE	结束矢量运动
VS 40000	进给率
BGS	开始圆弧运动
AMS	等待圆弧运动完成
PR, , 80000	抬起刀具 (Z轴)
BG Z	开始Z轴运动 (抬刀)
AM Z	等待Z轴运动完成
PR -21600	移动X轴
SP 20000	X轴速度
BGX	开始X轴运动
AM X	等待X轴运动完成
PR, , -80000	落下Z轴 (下刀)
BG Z	
AM Z	
CR 80000, 270, -360	第2个圆弧切割
VE	
VS 40000	
BG S	
AM S	
PR, , 80000	抬刀
BG Z	
AM Z	
VP -37600, -16000	X、Y轴返回到起点
VE	
VS 200000	
BG S	
AM S	
EN	

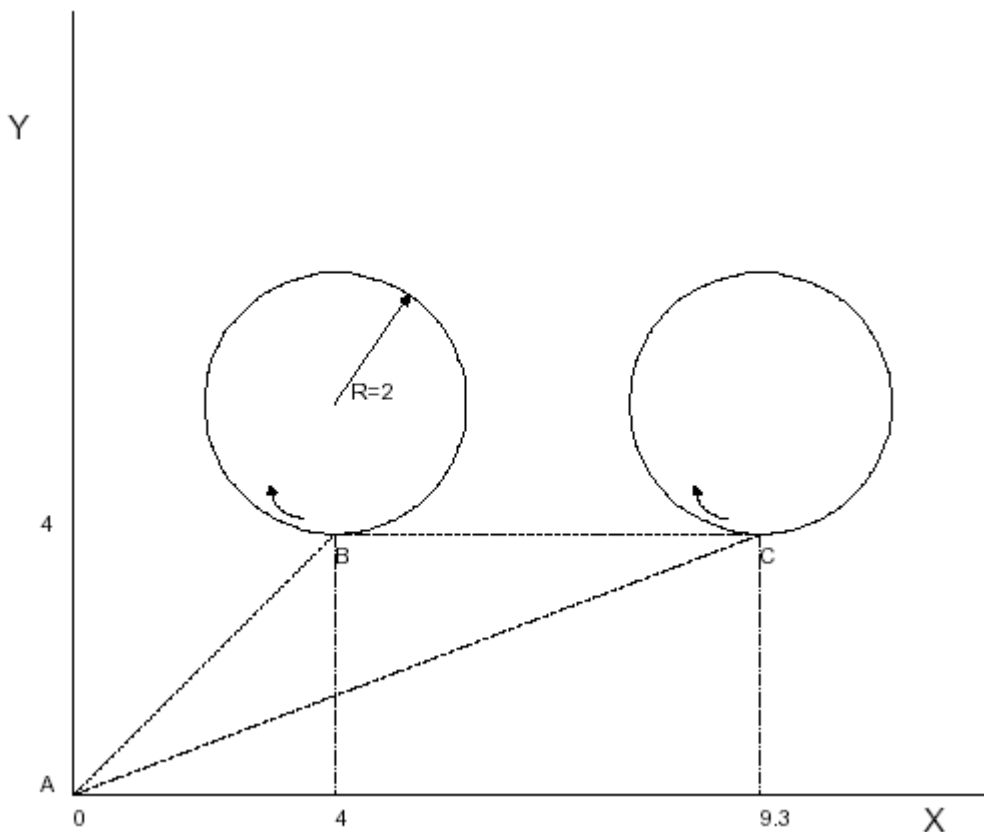


图7.2 – 圆弧、直线插补运动轨迹

15.3 用操控棒（Joystick）进行速度控制

用操控棒控制电机的速度，操控棒产生-10V~+10V之间的电压信号，目的是以与输入电压成比例的速度驱动电机运转。

假定，满10V电压对应电机速度为3000 rpm，编码器分辨率为1000线/rev即4000 cts/rev，此速度等于：

$$3000 \text{ rpm} = 50 \text{ rev/sec} = 200000 \text{ cts/sec}$$

程序周期性地读取输入电压，并将此值分配给变量VIN，要得到对应10V的速度200,000 cts/sec，我们选择速度为：

$$\text{Speed} = 20000 * \text{VIN}$$

将对应的电机速度分配给变量VEL。

指 令

A

JG0

BGX

B

VIN=@AN[1]

VEL=VIN*20000

JG VEL

JP # B

EN

15.4 用操控棒进行定位控制

此系统要求电机的位置与操控棒角度成比例，也就是说，两个位置之间比例必须是可编程的。例如，如果控制比例系数为5:1，就意味着，当操控棒电压为5V时，对应1024 cts，所要求的电机位置必须是5120 cts。变量V3改变位置比例系数。

指 令	说 明
#A	标号
V3=1024	初始位置比例系数
DP0	定义起始位置
JG0	设定JOG方式速度为0
BGX	开始运动
# B	
V1=@AN[1]	读模拟输入
V2=V1*V3	计算目标位置
V4=V2-_TPA-_TEA	求随动误差
V5=V4*20	计算比例速度
JG V5	改变JOG速度
JP # B	重复循环
EN	结束

15.5 用采样双闭环进行间隙补偿

连续双闭环由DB1函数进行使能，它是一种进行间隙补偿的有效方法。不过，在某些场合，当间隙较大时，就很难使系统稳定。此时，采用下述的采样双闭环方法就相对容易一些。

这里，针对运动控制系统中间隙所带来的麻烦，提出解决方案。目的是能精确地控制直线滑台的位置。滑台由旋转电机控制，电机通过丝杠机与滑台相连。这里，丝杠间隙为 $4\text{ }\mu\text{m}$ ，所要求的定位精度为 $0.5\text{ }\mu\text{m}$ 。

问题在于传感器安装在何处。如果采用旋转编码器，那么就存在 $4\text{ }\mu\text{m}$ 的间隙误差。另一方面，若采用直线编码器（光栅），由于不稳定性，反馈回路的间隙就会引起系统振荡。

另一种途径是采用双闭环，这里，我们采用两个传感器，旋转编码器和光栅。旋转编码器确保系统稳定性（因为在间隙之前进行位置闭环），而光栅提供精确的负载位置信息。其工作原理是驱动电机到终点附近的给定旋转位置。一旦到达，就读取负载位置，求出位置误差，控制器命令电机移动到消除此位置误差的新的旋转位置。

由于所要求的精度是 $0.5\text{ }\mu\text{m}$ ，光栅的分辨率应该高于其两倍。分辨率为 $0.25\text{ }\mu\text{m}$ 的光栅能得到 $\pm 2\text{ cts}$ 的位置误差。

双闭环方法要求旋转编码器的分辨率等于或好于直线编码器的分辨率。假定，丝杠螺距为 2.5mm （约10圈/inch），旋转编码器为2500线/rev，即10000cts/rev，即相当于 $0.25\text{ }\mu\text{m}/\text{cts}$ 脉冲当量，这样就使光栅和旋转编码器具有相同的分辨率。

为了说明这种控制方法，假定，旋转编码器用于X轴反馈，读取光栅反馈信号并存在变量LINPOS中，再假设，在起点处，X轴的位置和LINPOS的值都为0，现在，我们假定，要把直线负载移动到位置1000。

第一步是命令X轴电机移动到旋转位置1000，一旦到达，我们再检查负载位置，例如，若负载位置为980cts，这就意味着必须进行20cts的修正。不过，当把X轴移动到位置1000时，假设实际位置只时995，意味着X轴位置误差为5cts，一旦电机稳定下来，就会消除。这意味着只需要修正15cts，由于X轴已修正了20cts中的5cts。据此，运动修正就应该是：

修正值=负载位置误差-旋转位置误差

进行数次修正，直到误差小于 $\pm 2\text{cts}$ 。往往在一个修正循环中即可完成。

指 令	说 明
# A	标号
DP0	定义起始位置为0
LINPOS=0	
PR 100	距离
BGA	开始运动
# B	
AMA	等待运动完成
WT 50	暂停50 ms
LIN POS = _DEA	读直线位置
ER=1000-LINPOS-_DEA	求修正值
JP # C, @ABS[ER]<2	若误差小于2，则退出
PR ER	定义修正
BGA	
JP # B	重复循环
# C	
EN	

第八章 硬件保护和软件保护

1. 说明

DMC-2x00提供了几种硬件、软件保护功能，以诊断报警条件，并禁止电机运动。这些功能有助于保护各种系统元件免受损坏。

警告：机器在运动中可能有危险！用户有责任设计有效的报警处理及安全保护。由于DMC-2x00是机器中的集成部分，因此，工程师应该设计具有保护措施的系统，以免造成DMC-2x00元件损坏。对于任何伤害或损坏，GALIL不承担任何责任。

2. 硬件保护

DMC-2x00有硬件输入和输出保护线，用于报警和机械限位保护。

2.1 输出保护连线

1)。AMPEN（放大器使能）：

控制器给出关断电机命令（MO）时，此信号就变为低电平；还有其它一些条件都会使AMPEN变低，而切断使能，这些条件如下：

- （1）位置误差超过由ER命令（误差限制）所设定的限制值时（即位置超差时）；
- （2）当报警关断条件使能时（OE1）；
- （3）给出急停命令时。各轴放大器都有各自的放大器使能线（AMPEN），当看门狗定时器触发或复位时，此信号也变低从而关断伺服输出。

注意：AMPEN 信号的标准配置为TTL低有效。如果用户使用PICM-2900-OPTO或PICM-3900-OPTO接口板，就能对极性进行改变，由于在PICM-2900和PICM-3900中，对AMPEN输出信号已进行光隔处理，并采用OC输出，因此，可以与任何电压的伺服放大器或步进驱动器直接接口。

2)。报警输出：

在DMC-2x00及其它DMC控制器中，报警输出为TTL信号，以此表示控制器中的报警状态。此信号在互联模块（接口板）侧，称作ERROR。以下几种情况可能导致报警信号变低而出现报警输出：

- (1) 至少一个轴出现位置超差；误差限制值由ER命令进行设定。
- (2) 控制器侧的复位线一直为低或受干扰而影响。
- (3) 控制器有故障且处理器正在自身复位。
- (4) 驱动报警信号的输出IC芯片有故障。

注意：当用户选用PICM-2900-OPTO或PICM-3900-OPTO接口板与GALIL控制器配套时，报警输出信号ERROR已经光隔处理或OC输出形式，因此，可以很方便地与任何外部接口线路相连接。

2. 2 输入保护连线

1)。通用急停（ABORT）：

此输入信号一变低，就立即停止运动，而且是不经过减速处理的停止。对于报警关断功能使能的任何轴来说，都会使其使能信号（AMPEN）关断。这样会使电机依靠自身惯性自由滑行而停止。如果没有使报警关断功能使能，由于此时，控制器输出命令突然变为0，因此，电机立即停止，锁定在当前位置。

2)。选择性急停：

也可以对控制器进行配置，对各轴提供各自的急停。其作用与ABORT输入相同，但只对指定轴有效。要想把控制器配置成选择性急停，使用命令CN, , 1 即可。这样就将输入点 5, 6, 7, 8, 13, 14, 15, 16 配置成分别对应轴 A, B, C, D, E, F, G, H 的选择性急停。

3)。正向限位开关：

此信号变低禁止正向运动。当此限位开关生效时，电机正在以正方向运动，那么，就会使运动减速而停止。此外，如果电机正在以正方向运动，控制器就会自动跳转到限位开关子程序#LIMSWI(在用户将此子程序写入应用程序的情况下)。可以用CN命令改变限位开关的极性。

4)。反向限位开关：

此信号变低禁止反向运动。当此限位开关生效时，电机正在以反方向运动。那么，就会使运动减速而停止。此外，如果电机正在以反方向运动，控制器就会自动跳转到限位开关子程序#LIMSWI(在用户将此子程序写入应用程序的情况下)。可以用CN命令改变限位开关的极性。

3. 软件保护

DMC-2x00提供了可编程误差限制功能。用ERn命令设置误差限制值。其中：n= 1~32767。ER的缺省值为16384。

ER 200, 300, 400, 500 设置A轴误差限制为200, B轴为300, C轴为400, D轴为500
ER, 1, , 10 设置B轴误差限制为1cts, D轴为10cts.

误差限制值的单位为正交计数脉冲（cts）。误差是命令位置与实际编码器反馈值之差。如果误差的绝对值超过由ER所规定的值，DMC-2x00就会产生以下几种信号，警告主系统：

信号或功能	报警出现时的状态
# POSERR	跳转到超差自动处理子程序
Error Light	点亮
OE 功能	若为OE1，则关断电机使能
AEN 输出线	变低

设置条件语句进行跳转有助于根据不同报警在程序内部跳转。在运行期间，用TE命令可以对

A, B, C, D轴的位置误差进行监控。

3.1 可编程位置限制（软件限位）功能

DMC-2x00具有软件限位功能，可以用BL和FL软件命令对正、反向限位值进行软件设置。一旦指定位置限位值，DMC-2x00就不会接受超过此限制值的位置命令。当然，也防止了超过限制值的运动。

举例

指 令	说 明
DP0, 0, 0	定义位置
BL -2000, -4000, -8000	设定反向位置限制值
FL 2000, 4000, 8000	设定正向位置限制值
JG 2000, 2000, 2000	Jog运动
BG XYZ	开始
运行此程序，运动停止在正向限制值处	

3.2 报警关断功能（Off-On-Error）

DMC-2x00 控制器具有在某些报警条件下，关断电机的内置功能。此功能称之为“Off-On-Error”（报警关断）。要想使OE功能对各轴起作用，就指定OE1，要使此功能无效，就指定OE0。当此功能使能时，在以下3 种条件下，都会使指定的电机使能关断。.

- 1) . 指定轴的位置误差超过用ER命令设定的限制值；
- 2) . 给了急停命令（AB）
- 3) . 急停输入变低而起作用

注意：如果在电机运行期间，关断使能，那么，由于此时电机不再工作在伺服控制方式下，因此会依靠自身惯量而慢慢停止。

要想重新使系统使能，请使用复位（RS）或伺服使能（SH）命令。

举例：

OE 1, 1, 1, 1	对A、B、C、D轴，报警关断功能使能
OE 0, 1, 0, 1	对B、D轴报警关断功能使能，对A、C轴，报警关断功能无效

3.3 自动误差子程序

如果任何一个轴的误差超过了由命令ER所指定的误差极限值，# POSERR标号就使跟随其后的语句自动被执行。误差子程序必须用RE命令来结束。RE命令会使程序执行从误差子程序返回到主程序。

注意：如果不消除产生超差的条件，还会再次进入误差子程序。

举例：

指 令	说 明
#A; JP # A; EN	“虚拟” 程序
# POSERR	用超差起动误差子程序
MG “error”	发送信息
SB 1	输出点1置位（使继电器吸合）
ST A	停止A轴
AM A	在停止之后
SH A	伺服使能以清除误差
RE	返回主程序

注意：只有在应用程序执行期间，#POSERR子程序才会起作用。

3.4 限位开关子程序

DMC-2x00具有硬件正、反向限位功能。对于限位开关子程序，也有一个特殊标号，#LIMSWI标号指定开始限位开关子程序。如果任何一个限位开关起作用，而且该轴电机正在以同方向运动，那么此标号就使跟随其后的语句自动被执行。RE命令使此子程序结束。在条件跳转期间，也可以测试正、反向限位开关的状态。_LR指定反向限位，而_LF指定正向限位。跟随LR或LF的A，B，C，D指定轴。用CN命令配置限位开关的极性。

举例：

指 令	说 明
#A; JP # A; EN	虚拟程序
# LIMSWI	限位开关处理子程序
V1=_LFA	检查是否正向限位
V2=_LRA	检查是否反向限位
JP# LF, V1=0	如果处于正向限位，跳转到# LF
JP# LR, V2=0	如果处于负向限位，跳转到# LR
JP# END	跳转到#END
# LF	标号
MG “FORWARD LIMIT”	发送信息
STX; AMA	停止运动
PR-1000; BGA; AMA	反向运动
JP# END	跳转到#END
# LR	标号
MG “REVERSE LIMIT”	发送信息
STX; AMA	停止运动
PR1000; BG A; AM A	正向运动
# END	结束
RE	返回到主程序

注意：只有在应用程序执行期间，#LIMSWI才会起作用。

第九章 故障处理

1. 概述

对于一个完整的运动控制系统，涉及到的环节比较多，从硬件方面来讲，有与控制器相配套的互联模块、电缆、伺服电机或步进电机、伺服放大器或步进驱动器、反馈编码器以及与机械侧相连接的I/O及各部件之间的连接电缆等等。就软件而言，有系统参数设置，应用程序以及与之相配套的伺服放大器或步进驱动器的参数设定等等。因此，在使用中出现报警或发生故障，需要现场工程师冷静而认真分析，为了便于用户在使用现场快速诊断，我们将可能出现的问题分成下面几组：

- 1) 安装
- 2) 通信
- 3) 稳定性和补偿

4) 操作

我们将一些故障现象、引起原因及排除方法列于下表，供用户参考。

2. 安装

故 障 现 象	原 因	排 除 方 法
无附加输入，连接放大器时，电机飞车（暴走）	放大器偏置太大或正反馈	调整放大器偏置，检查连线
与上相同，但调整偏置后电机不停止	坏的放大器	更换放大器
控制器读不到编码器反馈值的变化	1 编码器连线错误	检查编码器连线
	2 坏的编码器	检查编码器信号，若需要，更换编码器
	3 坏的控制	将编码器连接到其它轴输入，若工作正常，则控制器故障，维修或更换控制器

3. 通信

故 障 现 象	原 因	排 除 方 法
用终端仿真器，不能与控制器通信	1 所选通信口不正确	试另一个通信口
	2 所选波特率不正确	确认波特率是否与控制器上的指拨开关设定相同，若需要，更改

4. 稳定性

故 障 现 象	原 因	排 除 方 法
当回路闭环时，电机暴走（飞车）	反馈极性不正确	通过转换电机电枢线（DC电机）或编码器，转换反馈极性
电机振荡	增益太大或阻尼常数（KD）太小	减小KP、KI，增大KD

5. 操作

故 障 现 象	原 因	排 除 方 法
控制器拒绝命令，以“？”回应	任何可能	用TC或TC1查询原因
电机没有完成运动	限位开关上的干扰信号使电机停止	要判断原因，检查停止码（SC）。若由限位开关干扰引起，减小干扰
在周期性运行期间，电机慢慢漂移	1 编码器干扰	重复查询位置，如果控制器读取的位置相差太大，则意味着编码器干扰。减小干扰，使用差分编码器输入信号

	2 编程错误	在运动结束点，避免用SH命令对位置误差进行复位
--	--------	-------------------------

第二篇 命令参考手册

第一章 概述

1. 控制器注解

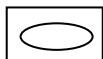
本篇是对《GALIL 用户手册》的补充说明。针对于某一特定型号的控制器操作，请参阅相应的用户手册。本命令手册适应于 GALIL 公司 Optima 系列如下型号的运动控制器：DMC-1200，DMC-1600，DMC-1700，DMC-1800，DMC-2000，DMC-2100 及 DMC-2200。本手册中所有命令按字母顺序进行解释说明。

请注意所有命令并非对于每一种控制器都有效。为了辨别各命令所适用的控制器，请查阅命令说明中的用途说明。

特别注意：GALIL 公司是当今世界上最著名的运动控制器生产厂家，截止 2002 年底，已有 30 余万台控制器在全世界可靠运行，产品多达 20 多个系列，本命令手册虽然针对高档 Optima 产品而展开说明，但除 A/D 转换输入、第二 FIFO、第二编码器检测输入等功能之外，其余功能命令说明对于其它系列产品（如 DMC18x2、DMC9542、DMC34x5、DMC14x5、DMC1417、DMC1410、DMC1412、DMC1416 等）仍然极具参考价值，原因在于 GALIL 任何一款控制器均具有相同的编程方法，在硬件功能相同的条件下，用户程序可以相互移植而运行。

2. 伺服电机和步进电机注解

GALIL 运动控制器既可以控制伺服电机，又可以控制步进电机，或者二者任意结合。因此，系统安装和参数配置就随您使用的是步进电机还是伺服电机而有所不同。为了使您能够快捷、方便地找到合适的命令，在具有特定适用类别的命令后面，标有特别警示标识符；若没有特别警示标识符，则该命令对伺服电机、步进电机均适用。特别警示标识符如下所示：



注意：适合于伺服电机使用。



注意：适合于步进电机使用。

3. 命令说明

每条可执行的命令按字母顺序在后续章节中列出，下面是针对每条命令所做说明的解释。

各条命令的二字符操作码置于右上角。有些命令有与之相对应的二进制码，此二进制码也在靠近 ASCII 命令旁在圆括号内列出。对于二进制命令方式，请见后续讨论。以下部分是对命令及所需定义的解释说明。

1) 轴定义

有些命令需要用户对所要控制的指定轴加以辨识，这些命令后面跟随有大写字母 X、Y、Z、W 或 A、B、C、D、E、F、G、H，各大写字母之间不需要逗号，轴的顺序也无关紧要。在任何命令之前不要插入空格。例如：STX; AMX 是无效命令，原因是在分号之前有空格。对于命令来说，正确的语句是在命令和定义之间用一个空格分开。在不需要定义或未给出定义时，此命令则对所有轴有效。

有效语句：

SHA	伺服使能，只对 A 轴有效
SHABD	伺服使能，对 A、B、D 轴有效
SHACD	伺服使能，对 A、C、D 轴有效
SHABCD	伺服使能，对 A、B、C、D 轴有效
SHBCAD	伺服使能，对 A、B、C、D 轴有效
SHADEG	伺服使能，对 A、D、E、G 轴有效
SHH	伺服使能，对 H 轴有效
SH	伺服使能，对所有使能

2) 参数定义

有些命令需要指令之后的数字加以定义。在定义描述中，这些命令之后跟随有小写 n，n，n，n，n，n，n，其中字母 n 代表数值，这些数值可以对任何一个轴分别指定，也可以对任意组合轴加以指定，各轴的定义使用逗号分开。有效语句举例如下：

有效语句

AC n	只对 a 轴指定数值
AC n, n	只对 a、b 轴指定数值
AC n,, n	只对 a、c 轴指定数值
AC n, n, n, n	对 a、b、c、d 轴指定数值
AC , n,,, n	只对 b、e 轴指定数值
AC ,,, n,, n	只对 d、f 轴指定数值

其中 n 由实际数值来取代。

3) 直接命令定义

定义数据的另一种方法是使用等号对所需要设置数据的轴进行数据设定。(*) 号可以置于轴指定的地方，(*) 号所指定的数据对所有轴有效。举例如下：

PRB=1000	将 B 轴移动量设置为 1000
PR*=1000	将所有轴的移动量设置为 1000

4) 问号

大多数命令可以接受问号 (?) 作为定义。此定义使控制器返回到命令说明中的参数处，在命令之后对所询问的轴键入?。语句格式与前面描述的参数定义相同，只不过是? 取代了数值。举例如下：

PR ?	让控制器返回 A 轴的相对位置值
PR ,,, ?	让控制器返回 D 轴的相对位置值
PR ? , ? , ? , ?	让控制器返回 A、B、C、D 轴的相对位置值
PR ,,,,,, ?	让控制器返回 H 轴的相对位置值

5) 操作用途

大多数命令有一个用于返回数值相对应的操作。操作用途描述就提供了这种语句以及用操作数返回的值。操作数必须使用在正确的 DMC 表达式里面。例如，要显示操作数值，用户就可以使用命令：

MG ‘Operand’

所有命令操作数均由下划线符()开始。例如：A 轴当前位置值能够用如下命令对变量“V”进行指定：

V=_TPA

6) 用途说明

用途说明规定了相关命令的用途。下面是所提供的命令信息的解释：

“While moving”：指控制器在执行运动当中，命令是否有效。

“In a program”：指命令是否可以作为用户定义程序的部分。

“Command line”：指命令是否可以用做直接命令。

“Controller usage”：辨别能够接收命令的控制器型号。

7) 缺省说明

在命令说明当中，DEFAULT 部分对控制器配置参数提供了缺省值。这些参数可以更改，更改后的参数值可以用命令“BN”保存在非易失存储器中。如果未将更改后的参数保存在非易失存储器中，当对系统进行复位后，参数仍旧不变，还是原有的缺省值。当电源关断后再上电，按下复位按钮，或执行 RS 命令均会视作复位。

8) 对控制器复位使参数恢复到出厂缺省值

当做主复位(Master reset)操作时，控制器就会将所有设置的参数恢复到出厂缺省值，而且将非易失存储器清除到出厂状态。用命令<Ctrl R><Ctrl S><Return>或者将 MRST 短路棒或拨码开关置于 ON 都会执行主复位操作。

例如：命令 KD 用于对各轴设置微分常数。微分常数的缺省值是 64，如不用命令 KD 设置此参数，对于各轴来说，控制器中仍然是 64，如果对微分常数 KD 做了更改，但未存储在非易失存储器中，在对控制器进行复位或重新上电后此项仍然是缺省值 64。如果对此项参数加以设置更改并存储在非易失存储器中，即使复位，而不是主复位，更改后的参数也将被保存。

缺省格式描述了命令提问时所返回的数值的格式。它代表小数点前、后的位数。

4. 二进制命令

某些命令针对控制器有等效的二进制值，这些值以十六进制格式列在靠近命令的圆括号内。二进制通信方式的速度比 ASC II 命令快得多。当从 PC 机发送命令且不能嵌入应用程序中时，只能使用二进制格式。

十六进制格式将一个字节表示为两个 4 位二进制数值。每个 4 位数值表示一个字符，此字符与十进制 0~15 之间的数值相对应，而代表 10~15 的字符分别是 A、B、C、D、E、F，例如，16 进制数 6D 的二进制数值为 01101101。负数以 2 的补码来表示。

1) 二进制命令格式

所有二进制命令均有一个 4 字节数领头，后面跟着数据域，此 4 字节数用十六进制格式表达。

领头格式：

字节 1 规定了 80~FF 之间的命令数，完整的二进制命令数表见本节第二部分。

字节 2 规定了各数据域中的字节号为 0, 1, 2, 4, 6, 如下所示：

- 00 非数据域（如 SH、BG）
- 01 每域一个字节
- 02 1 个字（每域 2 个字节）
- 04 每域 1 个长字（4 个字节）
- 06 GALIL 格式（4 字节整数+2 字节小数）

字节 3 规定命令是否用做坐标运动，如下所示：

- 00 无坐标运动控制
- 01 坐标运动控制

例如，命令 STS 指定矢量运动停止，等效二进制命令的第 3 字节应为 01。

字节 4 规定轴号或数据域，如下所示：

- Bit7=H 轴或第 8 个数据域
- Bit6=G 轴或第 7 个数据域
- Bit5=F 轴或第 6 个数据域
- Bit4=E 轴或第 5 个数据域
- Bit3=D 轴或第 4 个数据域
- Bit2=C 轴或第 3 个数据域
- Bit1=B 轴或第 2 个数据域
- Bit0=A 轴或第 1 个数据域

数据域格式：

数据域必须与格式字节和轴字节相一致。例如，命令 PR 1000,, -500 应为：

A7 02 00 05 03 E8 FE 0C

其中，A7 是命令 PR 的 16 进制数。

02 规定各数据域为 2 字节

00 S 对于 PR 无效

05 规定 Bit0 对 A 轴有效，Bit2 对 C 轴有效 ($2^0+2^2=5$)

03 E8 代表 1000

FE 0C 代表 -500

举例如下：

命令 ST ABC 表示为：A1 00 01 07

其中：A1 表示 ST 命令

00 表示数据域为 0

01 表示停止坐标轴 S

07 表示停止 A (Bit0)，B (Bit1)，C (Bit2) $2^0+2^1+2^2=7$

2) 二进制命令表

COMMAND	NO.	COMMAND	NO.	COMMAND	NO.
保留	80	保留	ab	保留	d6
KP	81	保留	ac	保留	d7
KI	82	保留	ad	RP	d8
KD	83	保留	ae	TP	d9
DV	84	保留	af	TE	da

AF	85	LM	b0	TD	db
KS	86	LI	b1	TV	dc
PL	87	VP	b2	RL	dd
ER	88	CR	b3	TT	de
IL	89	TN	b4	TS	df
TL	8a	LE,VE	b5	TI	e0
MT	8b	VT	b6	SC	e1
CE	8c	VA	b7	保留	e2
OE	8d	VD	b8	保留	e3
FL	8e	VS	b9	保留	e4
BL	8f	VR	ba	TM	e5
AC	90	保留	bb	CN	e6
DC	91	保留	bc	LZ	e7
SP	92	CM	bd	OP	e8
IT	93	CD	be	OB	e9
FA	94	DT	bf	SB	ea
FV	95	ET	c0	CB	eb
GR	96	EM	c1	II	ec
DP	97	EP	c2	EI	ed
DE	98	EG	c3	AL	ee
OF	99	EB	c4	保留	ef
GM	9a	EQ	c5	保留	f0
保留	9b	EC	c6	保留	f1
保留	9c	保留	c7	保留	f2
保留	9d	AM	c8	保留	f3
保留	9e	MC	c9	保留	f4
保留	9f	TW	ca	保留	f5
BG	a0	MF	cb	保留	f6
ST	a1	MR	cc	保留	f7
AB	a2	AD	cd	保留	f8
HM	a3	AP	ce	保留	f9
FE	a4	AR	cf	保留	fa
FI	a5	AS	d0	保留	fb
PA	a6	AI	d1	保留	fc
PR	a7	AT	d2	保留	fd
JG	a8	WT	d3	保留	fe
MO	a9	WC	d4	保留	ff
SH	aa	保留	d5		

5. 控制软件加速运行

运动控制器能够以快于伺服更新速率的方式运行。此方式称之为“fast mode”，允许的更新速率如下：

1~2 轴时	125usec
3~4 轴时	256usec
5~6 轴时	375usec
7~8 轴时	500usec

注：使用命令 TM 设置需要的更新速率

为了以“fast mode”运行运动控制器，必须上载快速控制软件。此项操作使用 GALIL 工具软件 DMCTERM 和 WSDK 来进行。使用“菜单选择”（menu option），“Update Flash EEPROM”来改变控制器中的控制软件。快速控制软件包含在控制器实用工具中。

以“fast mode”运行时，如下功能会无效或者被更改。

1) 在“fast mode”不允许运行的命令

命令
电子齿轮方式
ECAM 方式
极点 (PL)
模拟反馈 (AF)
步进电机运行 (MT 2, -2, 2.5, -2.5)
条件启动 (只允许域 0)
DMA 通道
速度响应命令 (TV)

2) 在“fast mode”运行时，被改变的命令

命令	更改
MT	命令定义 2, 2.5, -2, -2.5 无效
AD, AI, AM, AP, AR, AS, AT, AV, MC, MF, MR, WC	在域 1~7 中不允许的命令

6. 条件启动

Optima 系列控制器提供了几条可用来进行逻辑判断，即条件启动的命令，这些条件启动命令均以运行程序期间的事件为基础。这样的事件包括：规定的运动完成、等待要到达的某一位置或只是简单地等待一段时间。

控制器在执行程序时，各程序行按序执行。不过，当执行条件启动命令时，程序暂停执行下一行，只有条件启动的状态被清除，才会继续运行下一行。注意：条件启动只暂停执行所设置的域，而所有其它未被命令的域不受影响。此外，如果从子程序设置条件启动，就会使子程序及主域都暂停执行。

由于在程序运行期间，用条件启动命令作为程序流程指令，因此，不能从终端的命令直接实现这一功能。从终端的命令直接发送条件启动命令会引起主 PC 机和运动控制器之间的通信中断，直到条件启动被清除后，才会继续执行。

为了方便起见，下表列出了用于条件启动的命令及其基本用途：

AD	在距离到达之后启动
AI	在输入信号到达之后启动
AM	在运动完成后

AP	在绝对位置完成之后
AR	在相对位置完成之后
AS	速度到达
AT	时间到达
AV	在矢量距离到达之后
MC	定位完毕
MF	正向移动到某一个位置
MR	反向移动到某一个位置
WC	等待轮廓数据完成
WT	等待

7. 指令集

1) 自动子程序

以下自动子程序是 GALIL 已开发完成，可由用户直接使用的子程序

#AMPERR	(RE)	伺服放大器报警子程序
#AUTO	(EN)	上电后自动执行的程序
#AUTOERR	(EN)	若自检出现错误，上电后运行的报警程序
#CMDERR	(RE)	写有不正确的命令
#COMINT	(EN)	辅助串行口通信中断
#ININT	(RI)	由 II 变低所规定的输入
#LIMSWI	(RE)	坐标轴限位开关输入电平变低
#MCTIME	(RE)	运动完成超时，超时大小用 TW 命令设置
#POSERR	(RE)	位置误差超时 ER 所规定的上限
#TCPERR	(RE)	Ethernet 通信错误处理程序

2) 无刷电机命令

BA	无刷电机轴
BB	无刷电机相位起点
BC	无刷电机校准
BD	无刷电机相位角
BI	无刷电机相位检测输入
BM	无刷电机模数
BO	无刷电机偏置设定
BS	无刷电机初始配置
BZ	无刷电机零相位设置

3) 轮廓方式命令

CD	轮廓数据
CM	轮廓方式
DT	轮廓时间间隔
WC	等待轮廓数据

4) 电子凸轮 (ECAM) / 电子齿轮

EA	选择 ECAM 主动轴
EB	ECAM 使能
EC	ECAM 表索引
EG	ECAM 咬合
EM	ECAM 循环周期
EP	ECAM 间隔
EQ	ECAM 脱开
ET	ECAM 表入口
GA	设置电子齿轮主动轴
GM	龙门同步方式
GR	设置电子齿轮变比

5) 误差控制命令

BL	负向软件限位
ER	误差限制
FL	正向软件限位
OE	误差超限切断
TL	转矩限制
TW	定位超时

6) Ethernet 命令

AO	Modbus 装置的模拟输出电压
IA	设置 IP 地址
IH	Internet 端口
MB	Modbus
SA	发送命令
TH	响应端口状态
WH	端口辨认

7) I/O 命令

AL	锁存使能
CB	清除位
CI	通信中断
CO	配置 I/O 点
EI	中断使能
II	输入中断
OB	定义输出位
OC	输出比较功能
OP	发送数据到输出口
SB	设置位
UI	用户中断

8) 与运动相关的命令

AB	急停
AC	设置加速度
BG	运动开始
DC	设置减速度
FE	寻边
FI	寻找原点
HM	回零
HX	暂停执行
IP	增量位置
IT	平滑处理时间常数
JG	JOG 方式
PA	设置绝对位置
PR	设置相对位置
SP	设定速度
ST	停止执行

9) 查询命令

LA	显示数组名及大小
_LF	正向限位开关状态
LL	显示程序的标号
_LR	反向限位开关状态
LS	显示程序的指令内容
LV	显示变量名称及记忆值
MG	数据发送（向主机发送）命令
QR	记录数据
QZ	返回 DMA 信息
RP	读取命令位置
RL	读取锁存数据
^R^V	读取软件版本信息
SC	响应(传回,查询)电机停止的原因
TB	响应状态位
TC	响应错误码
TD	响应辅助 Encoder 位置
TE	响应误差
TI	响应输入点
TIME	时间操作，内部时钟
TP	响应位置
TR	响应轨迹
TS	响应开关
TT	响应转矩
TV	响应当前速度

10) 数字/函数运算命令

@SIN[X]	正弦计算
@COS[X]	余弦计算

@TAN[X]	正切计算
@COM[X]	补码计算
@ASIN[X]	反正弦计算
@ACOS[X]	反余弦计算
@ATAN[X]	反正切计算
@ABS[X]	取绝对值
@FRAC[X]	求余
@INT[X]	求整
@RND[X]	取整
@SQR[X]	平方根计算
@IN[X]	读取数字输入口[X]的状态
@OUT[X]	读取数字输出口[X]的状态

11) 程序编写命令

DA	定义数组大小
DL	下载程序
DM	定义数组
ED	程序编辑
ELSE	条件语句
ENDIF	结束条件语句
EN	程序结束
IF	条件语句
IN	输入变量
JP	跳转
JS	跳转到子程序
NO	作批注.无作用
RA	记录数据, 自动数据捕获
RE	从处理错误返回主程序
REM	对程序加标注
RI	从中断点返回主程序
UI	用户中断
UL	上载程序
XQ	执行程序
ZS	零堆栈

12) 伺服电机控制参数命令

AF	模拟反馈
FA	加速度前馈
FV	速度前馈
IL	积分极限
KD	微分常数
KI	积分常数
KP	比例常数
NB	凹陷滤波带宽

NF	凹陷滤波频率
NZ	凹陷滤波零点
OF	零点补偿
PL	低通滤波的 PID 补偿
SH	激活伺服状态(servo on)
TL	转矩极限
TM	采样时间

13) 步进电机控制命令

DE	定义辅助编码器的位置
DP	定义位置
KS	步进电机平滑处理
MT	电机类型选择
RP	返回命令位置
TD	响应辅助编码器位置
TP	响应编码器位置

14) 系统配置命令

BN	将参数烧录入 Flash 中
BP	将程序烧录入 Flash 中
BV	将变量和数组烧录入 Flash 中
CC	配置串行通信口 2
CE	配置编码器类型
CF	配置缺省通信口
CN	配置回原点、限位、锁存输入开关的电平极性
CW	数据调整开/关
DE	定义辅助编码器位置
DP	定义位置
DR	DMA/FIFO 读取速率
DV	双回路控制阻尼
EI	中断使能
EO	回声关断
IT	独立时间常数（平滑处理用）
LZ	将有效值之前的零点移除
MO	电机关断
MT	定义电机类型
PF	位置格式
QD	下载数组
QU	上载数组
RS	程序复位
^R^S	主控制程序复位
VF	定义变量格式

15) 条件启动命令

AD	在距离到达之后启动
AI	在输入信号到达之后启动
AM	在运动完成后
AP	在绝对位置完成之后
AR	在相对距离完成之后
AS	速度到达
AT	时间到达
AV	在矢量距离到达之后
MC	在运动完成，且到达位置之后启动
MF	在运动方向变正之后启动
MR	在运动方向变负之后启动
WC	在轮廓数据完成之后启动
WT	在等待时间到达之后启动

16) 矢量/直线插补命令

CA	定义坐标系
CR	圆弧插补运动
CS	清除矢量程序
ES	椭圆比例缩放
LE	直线插补结束
LI	直线插补距离
LM	直线插补方式（最多 8 轴）
ST	停止运动
TN	正切运动
VA	设定矢量加速度
VD	设定矢量减速度
VE	矢量程序结束
VM	坐标运动方式
VP	矢量位置
VR	矢量速度比
VS	矢量速度
VT	平滑处理时间常数（矢量方式）

第二章 指令详解

1) AB（二进制 A2）

功能：急停

说明：AB 命令使运动在没有减速控制的情况下立即停止。如果有程序正在运行，且未指定 1 参数，AB 也使程序紧急停止。对于报警关断功能使能的各轴来说，AB 命令将使电机使能关断。（详见命令“OE”说明）

AB 命令使运动中的所有轴紧急停止运动，不能用它停止个别轴。

参数：ABn

其中: n=0 控制器紧急停止运动和程序

n=1 控制器只停止运动

无后缀参数将使控制器紧急停止运动和程序。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	-
缺省格式	-

操作应用:

_AB 读取所给急停输入的状态, 1 无效, 0 有效

相关命令:

“SH (二进制 AA)”	激活伺服状态(servo on)
“OE (二进制 8D)”	位置错误关断电机

举例:

AB	停止运动
OE 1, 1, 1, 1	位置错误时, 使能关断功能有效
AB	关断电机命令并停止运动
#A	程序标号—程序开始
JG 2000	指定 X 轴 JOG 速度
BGX	X 轴开始 JOG 运动
WT 5000	等待 5000msec
AB1	停止运动, 但不停程序
WT 5000	等待 5000msec
SH	激活伺服状态
JP #A	跳转到标号#A
EN	程序结束

提示: 请记住, 如果您只想使运动紧急停止, 就要在 AB 后面使用参数 1。否则, 您的应用程序也将被紧急停止。

2) AC (二进制 90)

功能: 设置加速度

说明: AC 命令对独立运动 (如 PR、RA、JG 等) 设置电机的线性加速度。在运动期间, 可以改变加速度。用与之相对应的 DC 命令来设置减速度。

参数: AC n, n, n, n, n, n, n, n 或 ACA=n

其中: n 是 1024~67107840 之间的无符号数。输入的参数将被取整, 其最小系数为 1024。此参数的单位为计数单位/sec²。

n=? 返回指定轴的加速度值。

用途:

在运动当中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	25600
缺省格式	8.0

操作应用:

_ACX 读取指定轴的加速度值。

相关命令:

“DC (二进制 91)”	设置减速度
“FA (二进制 94)”	前馈加速度
“IT (二进制 93)”	平滑时间常数—S 曲线

举例:

AC 150000, 200000, 300000, 400000	设置 A 轴加速度为 150000 计数单位/sec ² B 轴为 200000 计数单位/sec ² C 轴为 300000 计数单位/sec ² D 轴为 400000 计数单位/sec ²
AC ? , ? , ? , ?	查询加速度
0149504, 0199680, 0299008, 0399360	显示返回的加速度值 (分辨率 1024)
V=_ACB	将 B 的加速度分配为变量 V

提示: 请根据您的传动系统及负载条件设置合适的加速度, 如电机转矩、负载、驱动器输出电流大小等。加速度过大会使加速期间的跟踪误差增大, 电机不能很好地跟踪命令轨迹。使用加速度前馈命令 FA 将会减小跟踪误差。

3) AD (二进制 CD)

功能: 在距离到达之后启动

说明: AD 命令是用来控制事件时序的条件启动命令。此命令将使后续命令的执行挂起, 直到满足下述条件之一才会继续执行。

- 命令的电机位置越过了运动开始时所规定的相对距离。
- 坐标轴运动轨迹完成。
- 命令的运动方向与规定位置的方向不一致。

此命令的单位是 4 倍频计数。1 次只能指定一个轴。运动轨迹必须有效, 否则条件启动将会自动满足。

注: 当运动平滑处理进间常数 IT 不等于 1 时, 将会影响 AD 命令。详见 IT 命令。

参数: AD nn,n,n,n,n,n,n 或 ADA=n 或 AND=n

其中: n 是 0~2147483647 之间的无符号整数

AND= 对 N 轴设置条件启动

注: 每条 AD 命令只能加注 1 个参数。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	-
缺省格式	-

相关命令:

“AV”	在矢量距离到达之后
“AP (二进制 CE)”	在绝对位置到达之后
“AR (二进制 CF)”	在相对距离到达之后
“MF (二进制 CB)”	正向移动到某一个位置
“MR (二进制 CC)”	反向移动到某一个位置

举例:

#A; DP0, 0, 0, 0	程序开始
PR10000, 20000, 30000, 40000	定义相对距离

BG	开始运动
AD 5000	在 A 轴到达 5000 之后
MG “Halfway to A”; TPA	发送信息
AD, 10000	在 B 轴到达 10000 之后
MG “Halfway to B”; TPB	发送信息
AD,, 15000	在 C 轴到达 15000 之后
MG “Halfway to C”; TPC	发送信息
AD,,, 20000	在 D 轴到达 20000 之后
MG “Halfway to D”; TPD	发送信息
EN	程序结束

提示：AD 命令精确到 2msec 内所获得的计数值。用速度乘以 2msec 得到最大位置误差。请记住：AD 从一个轴的运动起点计量相对距离。

4) AF (二进制 B5)

功能：模拟电压反馈

说明：模拟反馈 (AF) 命令用来将该轴设置成模拟电压反馈方式以取代数字反馈 (正交脉冲或脉冲+方向)。模拟反馈经由 12bit ADC 进行译码，因此，10V 输入电压对应 2047cts，-10V 输入电压对应-2048cts。16bit ADC 作为选择功能，用户在订货时需要特别指定，当为 16bit ADC 时，10V 输入电压对应 32768cts 的位置，而-10V 输入电压对应-32767cts 位置值。

参数：AF n,n,n,n,nn,n,n 或 AFx=n

其中：n=1 模拟反馈使能
n=0 模拟反馈无效，并切换到数字反馈
n=? 读取所指定轴模拟反馈状态，0：无效，1：使能

用途：

运动进行中	No	缺省：	缺省值	0, 0, 0, 0
在程序中	Yes		缺省格式	-
命令行	Yes			
适用控制器	所有型号控制器 (高档控制器及 DMC14x5/34x5)			

操作应用：

如果所指定轴模拟反馈使能，_AFx 返回值为 1，否则为 0。

相关命令：

“MT”	电机类型
“CE”	配置编码器

举例：

AF1, 0, 0, 1	X、W 轴模拟反馈
V1=_AFx	将反馈类型分配给变量 V1
AF? , ? , ?	查询反馈类型

5) AI (二进制 D1)

功能：在输入信息到达之后启动

说明：AI 命令是用于运动程序中等待直到指定的输入点改变状态之后，才运行后续命令的条件启动命令。

参数：AI +/-n

其中：n 是 1~96 之间的整数，代表输入点序号。若 n 为正，则控制器等待此输入点状态变高。若 n 为负，则等待此输入点状态变低。

用途：	缺省：
运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

相关命令：	
@IN[n]	用来读取输入点 1~8
“II（二进制 EC）”	输入中断
#ININT	输入中断标号

举例：	
#A	程序开始
AI8	等待输入点 8 变高
SP10000	指定速度为 10000 计数单位/sec
AC20000	指定加速度为 20000 计数单位/sec ²
PR400	指定运动位置
BGA	开始运动
EN	程序结束

提示：AI 命令实际上暂停执行直到所指定的输入满足所需的逻辑电平。如果您不要求程序按序暂停，请使用条件跳转命令（JP）或输入中断（II）命令。

6）AL（二进制 EE）

功能：锁存使能

说明：AL 命令使控制器的锁存功能（高速主编码器位置或辅助编码器位置捕获）有效。位置锁存功能有效时，该点信号变低，此时，主、辅编码器位置将被捕获。每个控制轴均有一个位置锁存输入，与如下通用输入点相对应：

A 轴锁存	Input1
B 轴锁存	Input2
C 轴锁存	Input3
D 轴锁存	Input4
E 轴锁存	Input9
F 轴锁存	Input10
G 轴锁存	Input11
H 轴锁存	Input12

命令 RL 返回指定轴的捕获位置值。若对某坐标轴锁存使能，且对 AL 命令查询时，返回值将为 1，而在锁存发生后，返回值将为 0。可以用 CN 命令来改变锁存函数的极性。

参数：AL nnnnnnnn 或 AL n,n,n,n,n,n,n,n

其中：n 可以是 A，B，C，D，E，F，G，H。n 的值用于定义欲锁存轴所指定的主编码器。

n 可以是 SA，SB，SC，SD，SE，SF，SG，SH。n 的值用于定义欲锁存轴所指定的辅助编码器。

用途：	缺省：
------------	------------

运动进行中	Yes	缺省值	0
在程序中	Yes	缺省格式	1.0
命令行	Yes		
适用控制器	所有型号控制器		

操作应用：

_Aln 读取指定锁存的状态。0=锁存无效，1=锁存使能

相关命令：

“RL（二进制 DD）” 读取锁存值

举例：

#START	起动程序
ALB	B 轴锁存使能
JG, 50000	设置 JOG 速度为 50000 计数单位/sec
BGB	开始运动
#LOOP	循环执行直到锁存出现
JP#LOOP, _ALB=1	
RLB	传送锁存的位置
EN	程序结束

7) AM（二进制 C8）

功能：在运动完成后

说明：AM 命令是一个条件启动命令，用于控制事件的时序。此命令将使后续命令的执行挂起，直到该轴当前运动完成。可以用 AM 命令对任意轴的组合或运动顺序加以定义。例如，AMAB 等待 A 和 B 两个轴运动完成。不带参数的 AM 命令表示等待所有轴的运动完成。

参数：AM nnnnnnnn

其中：n 是 A, B, C, D, E, F, G, H, S, T 或指定轴或 I/O 事件的任意组合。对于虚拟轴 N 轴来说，n 用 N 来表示。若 AM 命令后面没有带参数，表示等待所有轴的运动或 I/O 事件完成之后。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes*
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	1.0

* 对于 Ethernet 控制器来讲，此项无效。

相关命令：

“BG（二进制 A0）” 若运动完成，_BGn 返回值为 0。

举例：

#MOVE	程序名为 MOVE
PR5000, 5000, 5000, 5000	相对距离运动
BGA	启动 A 轴
AMA	A 轴运动完成之后
BGB	启动 B 轴
AMB	B 轴运动完成之后

BGC	启动 C 轴
AMC	C 轴运动完成之后
BGD	启动 D 轴
AMD	D 轴运动完成之后
EN	程序结束

提示：对于控制多轴运动顺序之间的时序来讲，AM 是一个非常重要的命令。例如，假如 A 轴处于增量运动距离（PR）的中间位置，只有在第一个运动完成之后，您才能进行绝对位置运动（PAA，BGA）。使用 AMA 命令，只有在第一个运动完成之后，才能暂停程序。AM 检测运动轨迹（包络线）完成。实际的电机仍可能在运动中。检测运动完成的另一种方法是检查内部变量_BGn，确认其值等于 0（见“BG”命令）。

8) AO

功能：模拟输出

说明：AO 命令设定由 Ethernet 相连接的 Modbus 装置的模拟输出电压。

参数：AO M, N

其中：m 是 I/O 编号，用下列公式计算：

$$m = (\text{SlaveAddress} * 10000) + (\text{HandleNum} * 10000) + ((\text{Module}-1) * 4) + (\text{BitNum}-1)$$

当 ModBus 装置有从装置与其相连时，使用 Slave address 可指定范围为 0~255。请注意，ModBus 挂用从装置的情况非常罕见，因此，此值通常为 0。

Handle Num 是端口指定器，从 A 到 F。

Module 是机柜中模块的位置，从 1~16。

BitNum 是模块中的 I/O 点，从 1~4。

n= 从+9.99~-9.99 之间的电压值。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes

缺省：

缺省值	-
缺省格式	-

适用控制器 只适用于 DMC-2100 系列

相关命令：

“SB（二进制 EA）”	设置输出位
“CB（二进制 EB）”	清除输出位

9) AP（二进制 CE）

功能：在绝对位置完成之后

说明：AP 命令是条件启动命令，用来控制事件的时序。此命令将使后续命令的执行挂起，直到满足下列之一条件：

- 实际电机位置越过指定的绝对位置。使用步进电机时，步进指令输出值（由输出缓存器来决定）越过了指令位置，即满足了这一条件。进一步的说明参见《用户手册》第 6 章“步进电机运行”。
- 坐标轴的运动轨迹（包络线）完成。
- 命令的运动方向与指定轴的位置方向不一致。

此命令值的单位是 4 倍频计数单位。一次只可指定一个轴。运动轮廓必须处于有效，否则，条件启动将会自动满足。

参数：AP nn,n,n,n,n,n,n 或 APA=n

其中：n 是-2147483648~+2147483647 之间的十进制符号整数。

用途：		缺省：	
运动进行中	Yes	缺省值	-
在程序中	Yes	缺省格式	-
命令行	Yes		
适用控制器	所有型号控制器		
相关命令：			
“AD（二进制 CD）”		相对距离条件启动	
“MF（二进制 CB）”		正向运动条件启动	

举例：		
#TEST		程序名
DP0		定义工作零点
JP1000		指定 JOG 速度 1000 计数单位/sec
BGA		开始运动
AP2000		在通过位置 2000 之后
V1=_TPA		分配变量 V1
MG “Position is”, V1=		打印信息
ST		程序停止
EN		程序结束

提示：AP 命令精确到 2msec 内的计数值。用 2msec 乘以速度来得到最大误差。AP 检测绝对位置。用 AD 命令来测量增量距离。

10) AR（二进制 CF）

功能：在相对距离完成之后

说明：AR 命令是用来控制事件时序的条件启动命令。此命令将后续命令的执行挂起，直到满足下列之一条件：

- 电机实际位置从运动起点或上一条 AR 命令或 AD 命令开始越过指定的相对距离。
使用步进电机时，步进指令输出值（由输出锁存器来决定），越过了指定的相对位置，即满足了这一条件。进一步的说明参见《用户手册》第 6 章“步时电机运行”。
- 坐标轴的运动轨迹（包络线）完成。
- 命令的运动方向与指定位置的方向不一致。

此命令值的单位是 4 倍频计数单位。一次只可指定一个轴。运动轮廓必须处于有效，否则条件启动将会自动满足。

注：当运动平滑时间常数 IT 不为 1 时，将会影响 AR。详细信息参见 IT 命令。

参数：AR nn,n,n,n,n,n,n 或 ARA=N

其中：n 是 0~2147483647 之间的十进制无符号整数。

用途：		缺省：	
运动进行中	Yes	缺省值	-
在程序中	Yes	缺省格式	-
命令行	Yes		
适用控制器	所有型号控制器		

相关命令：		
“AV”		用于联动控制矢量位置到达之后的条件启动
“AP”		用于绝对位置到达之后的条件启动

举例：

#A; DP 0, 0, 0, 0	程序开始
JG 50000,,, 7000	设置 JOG 速度
BGAD	开始运动
#B	标号
AR25000	A 轴越过 25000 计数单位相对距离之后
MG "Passed_A"; TPA	发送 A 轴信息
JP #B	跳转到标号#B
EN	程序结束

提示：用 AR 来指定从上一条 AR 或 AD 命令算起的增量距离。如果在单个运动程序中需要多个位置条件启动，就使用 AR。

11) AS (二进制 DO)

功能：速度到达

说明：AS 命令是一个条件启动命令，出现在实际运动速度到达指定速度时。此命令将使后续命令的执行挂起，直至到达指定速度。AS 命令在加速或减速之后才会工作。若速度未到达，在运动停止之后（减速之后）也会触发条件启动。

参数：AS nnnnnnnn

其中：n 是 A, B, C, D, E, F, G, H, S 或 T，或者指定坐标轴或 I/O 顺序的任意组合。对于虚拟轴 N 来说，n 是 N。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	-

举例：

#SPEED	程序名
PR100000	设置位置值
SP10000	设置速度值
BGA	开始 A 轴运动
ASA	A 轴速度到达之后
MG "At Speed"	输出信息
EN	程序结束

警告：AS 命令只适用于有线性加速度的梯型包络线速度运动，若用于有平滑处理的运动，不是很精确。

12) AT (二进制 D2)

功能：时间到达

说明：AT 命令是一个条件启动命令，用来使下一条命令的执行挂起直到所指定的时间消逝。常用此命令来测量时间。AT0 建立初始参考值，Atn 指定从参考值起计 n 毫秒 (msec)。AT-n 指定从参考值起计 n 毫秒，并在消逝的时间周期之后建立起新的参考值。

参数：Atn

其中：n 是 $0 \sim 2 \times 10^8$ 之间的有符号偶整数。

n=0 定义以当前时间为参考的时间

$n > 0$ 表示从参考时间开始等待 n 毫秒。

$n < 0$ 表示从参考时间开始等待 n 毫秒并当满足条件启动时重新设置参考时间。

($At-n$ 等效于 Atn ; $AT < \text{旧参考时间} + n >$)

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	0
缺省格式	-

举例:

按序发送如下命令

AT 0	建立参考时间 0 作为当前时间
AT 50	从参考 0 开始等待 50msec
AT 100	从参考 0 开始等待 100 msec
AT -150	从参考 0 开始等待 150 msec, 并设定新参考时间为 150
AT80	从新参考时间开始等待 80 msec (总等待时间为 230 msec)

13) AV

功能: 在矢量距离到达之后

说明: AV 命令是一个条件启动命令, 在联动控制期间如 VP、CR 或 LI, 用来使下一条命令的执行挂起。此条件启动应用在程序的轨迹距离到达所指定的值时。用此命令测量从联动起点开始或从上一条 AV 命令开始的距离。此命令的单位是 4 倍频计数单位。

参数: AV s, t

其中: s 和 t 是 0~2147483647 之间的十进制无符号整数。“s”代表在 S 坐标系中执行的矢量距离, “t”代表在 T 坐标系中执行的矢量距离。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	0
缺省格式	-

操作应用:

_AVS 读取 S 坐标系中从程序起点开始的矢量距离, _AVT 读取 T 坐标系中从程序起点开始的矢量距离。

举例:

#MOVE; DP 0, 0	程序标号
CAT	指定 T 坐标系
LMAB	A, B 轴直线插补运动
LI 1000, 2000	指定运动距离
LI 2000, 3000	指定运动距离
LE	
BGT	开始在 T 坐标系中运动
AV , 500	在轨迹距离等于 500 之后
MG “path>500”; TPAB	输出信息
EN	程序结束

提示: 在直线插补或矢量方式中, 用各轴距离的平方之和的平方根来计算矢量距离。

14) BA

功能：无刷电机轴

说明：BA 命令为正弦波换向电机配置控制器轴，并重新配置控制器以反映能够实际控制的电机数量。每个正弦波换向电机轴需要 2 个电机命令信号。第二路电机命令信号将始终与控制器中的最高轴相配合。例如，为具有 A 和 C 的 3 轴控制器配置正弦波换向将需要 5 路指令输出（5 轴控制器），其中 A 和 C 的第二路输出分别为 D 轴和 E 轴。

参数：BA xxxxxxxx

其中，n 是 A, B, C, D, E, F, G 或用来为正弦波换向无刷电机轴指定坐标轴的任意组合。无后缀参数表示取消为正弦波换向而配置的所有轴。

用途：

运动进行中	No
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	0

操作应用：

_Ban 读取所选正弦波换向轴的次相位的辅助 DAC 的轴号。对于 A 轴 DAC 而言，轴号以 0 开始。若将电机配置为标准伺服电机或步进电机，_BAn 读取值则为 0。

相关命令：

“BB（二进制 9E）”	无刷电机相位起点
“BC”	无刷电机换相
“BD（二进制 9D）”	无刷电机角度
“BI”	无刷电机输入
“BM（二进制 9B）”	无刷电机模数
“BO（二进制 9F）”	无刷电机偏置
“BS”	无刷电机设定
“BZ”	无刷电机零点

15) BB（二进制 9E）

功能：无刷电机相位起点

说明：BB 函数说明霍尔传输点与 $\theta=0$ 之间的位置偏置，用于正弦波换向电机。此命令必须保存在非易失性存储器中以便在复位时有效。

参数：BB n,n,n,n,n,n,n,n 或 BAA=n

其中：n 是带符号整数，代表被选轴的相位偏置，以 30° 的倍频表示。
n=? 返回指定轴的霍尔偏置值。

用途：

运动进行中	No
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	0

举例：

BB 30,, 60 分别表示 Y、W 轴的偏置值为 30° 和 60° 。

操作应用：

_BBn 读取指定轴霍尔传输点和 $\theta = 0$ 之间的位置偏值。

相关命令：

“BA”	无刷电机轴
“BC”	无刷电机换向
“BD (二进制 9D)”	无刷电机角度
“BI”	无刷电机输入
“BM (二进制 9B)”	无刷电机模数
“BO (二进制 9F)”	无刷电机偏置
“BS”	无刷电机设定
“BZ”	无刷电机零点

注：BB 只有作为 BC 命令的部分或复位时有效。

16) BC

功能：无刷电机校准

说明：函数 BC 监控正弦波换向电机霍尔传感器的状态，并在检测到第一个霍尔传感器信号时使换相相位复位。以此用由霍尔传感器所确定的更为精确的值取代预估换相相位值。

参数：BC nnnnnnn

其中：n 是 A, B, C, D, E, F, G 或所要指定轴的任意组合。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	0

操作应用：

_BCn 读取霍尔传感器输入状态，此值应在 1~6 之间。

相关命令：

“BA”	无刷电机轴
“BB (二进制 9E)”	无刷电机相位始点
“BD (二进制 9D)”	无刷电机相位角
“BI”	无刷电机输入
“BM (二进制 9B)”	无刷电机模数
“BO (二进制 9F)”	无刷电机偏置
“BS”	无刷电机设置
“BZ”	无刷电机零点

17) BD (二进制 9D)

功能：无刷电机相位角

说明：此命令设定正弦波换向电机的换向相位。用霍尔传感器时，通过使用命令 BC 能够对此参数设定更精确的值。只有在用户创建特殊的相位初始化时才可使用此命令。

参数：BD n,n,n,n,n,n 或 BDA=n

其中：n 是 0~360° 之间的整数。

n=? 返回无刷电机当前相位角 (0~360° 之间)

用途：

运动进行中	No
-------	----

缺省：

缺省值	0
-----	---

在程序中	Yes	缺省格式	0
命令行	Yes		
适用控制器	所有型号控制器		

操作应用：

_BDn 读取指定轴的换向相位角。

相关命令：

“BA”	无刷电机轴
“BB（二进制 9E）”	无刷电机相位始点
“BD（二进制 9D）”	无刷电机相位角
“BI”	无刷电机输入
“BM（二进制 9B）”	无刷电机模数
“BO（二进制 9F）”	无刷电机偏置
“BS”	无刷电机设置
“BZ”	无刷电机零点

18) BG（二进制 A0）

功能：开始执行程序

说明：BG 命令起动所指定轴或 I/O 程序的执行运动。

参数：BG nnnnnnnn

其中：n 是 A, B, C, D, E, F, G, H, N, S 或 T, 或者所指定轴的任意组合。

用途：

运动进行中	Yes	缺省值	0
在程序中	Yes	缺省格式	0
命令行	Yes		
适用控制器	所有型号控制器		

缺省：

操作应用：

如果所指定轴或坐标系完成运动，_BGn 读取 “0”，否则为 “1”。

相关命令：

“AM（二进制 C8）”	在运动完成之后
“ST（二进制 A1）”	停止运动

举例：

PR 2000, 3000, , 5000	设定 A、B、D 轴的相对运动距离
BG ABD	A、B、D 轴电机开始运动
HM	设定回零
BGA	只有 A 轴开始运动
JG 1000, 4000	设定 A、B 轴 jog 速度
BGB	只有 B 轴开始运动
BSTATE=_BGB	若 B 轴完成运动，分配 1 给 BSTATE
VP 1000, 2000	指定矢量位置
VS 20000	指定矢量速度
BGS	开始联动程序
VMAB	矢量模式
VP4000, -1000	指定 A、B 轴的矢量位置
VE	矢量模式结束

PR , , 8000, 5000 指定 C、D 轴的位置
 BGSCD 程序开始, C、D 轴运动
 MG _BGS 若运动发生在“S”坐标系中, 显示 1

提示: 当运动未完成时, 任意轴都不能执行 BG 命令。在运动期间, 可使用 AM 条件启动去等待运动完成。同时, 也可以通过读取_BGn 值来检测运动完成。

19) BI

功能: 无刷电机输入

说明: 当霍尔传感器用于正弦波换向电机相位检测时, 用 BI 命令来定义所使用的输入。这些输入可以是通用输入 (Bits1~8), 辅助编码器输入 (Bits81~96), 或者扩展 I/O 输入 (Bits17~80)。各轴的霍尔传感器必须连接到相应的输入连线上, 例如: BI3 指出输入 3, 4 且输入 4 用于霍尔传感器。

可以用无刷电机设置命令 BS 来确认霍尔传感器的连线是否正确。

参数: BI n,n,n,n,n,n,n,n 若 BIA=n

其中: n 是无符号整数, 它代表用于霍尔传感器输入的首位数字输入。

n=0 清除该轴霍尔传感器配置。

n=? 返回指定轴用于霍尔传感器的起始输入点。

用途:

运动进行中 Yes
 在程序中 Yes
 命令行 Yes

适用控制器 所有型号控制器

缺省:

缺省值 0
 缺省格式 0

操作应用:

_Bin 读取指定轴用于霍尔传感器的起始输入点状态。

相关命令:

“BA”	无刷电机轴
“BB (二进制 9E)”	无刷电机相位始点
“BC”	无刷电机校准
“BD (二进制 9D)”	无刷电机相位角
“BM (二进制 9B)”	无刷电机模数
“BO (二进制 9F)”	无刷电机偏置
“BS”	无刷电机设置
“BZ”	无刷电机零点

举例:

BI , 5 Y 轴的霍尔传感器占用输入 5, 6, 7 三个点

20) BL (二进制 8F)

功能: 负向软限位

说明: BL 命令设定负向软限位值。如果在运动期间, 运动距离超过此限位值, 该轴运动就会减速停止, 不允许有超越此限位值的负向运动。

如果自动子程序#LIMSWI 写入程序中并执行程序, 当负向软限位值被超越时, 就会执行该自动子程序#LIMSWI。

参数: BL n, n, n, n, n, n, n, n 或 BLA=n

其中: n 是-2147483648~2147483647 之间的带符号整数, 负向限位在位置 n-1 处被激活,

其单位是 4 倍频计数单位。

n=-2147483648 关掉负向软限位功能

n=? 返回指定轴的负向软限位值。

用途:

运动进行中 Yes

在程序中 Yes

命令行 Yes

适用控制器 所有型号控制器

缺省:

缺省值 -2147483648

缺省格式 位置格式

操作应用:

_BLn 读取指定轴的负向软限位值。

相关命令:

“FL (二进制 8E)” 正向软限位

“PF” 位置格式

举例:

#TEST	程序名
AC 1000000	设定加速度
DC 1000000	设定减速度
BL -15000	设定负向软限位值
JG -5000	设定负向 JOG 运动
BGA	开始运动
AMA	在运动完成后 (限位出现)
TPA	读取位置值
EN	程序结束

提示: GALIL 控制器也提供硬限位功能。

21) BM (二进制 9B)

功能: 无刷电机模数

说明: BM 命令以编码器计数单位定义磁极周期长度。

参数: BM n,n,n,n,n,n,n,n,n 或 BMA=n

其中: n 是 1~1000000 之间的十进制值, 分辨率为 1/10, 此值也能够指定为分辨率为 1/16 的小数值。

n=? 返回指定轴的无刷电机系数。

用途:

运动进行中 No

在程序中 Yes

命令行 Yes

适用控制器 所有型号控制器

缺省:

缺省值 0

缺省格式 0

操作应用:

_BMn 读取指定轴的周期长度值。

相关命令:

“BA” 无刷电机轴

“BB (二进制 9E)” 无刷电机相位始点

“BC” 无刷电机校准

“BD (二进制 9D)” 无刷电机相位角

“BI”	无刷电机输入
“BO（二进制 9F）”	无刷电机偏置
“BS”	无刷电机设置
“BZ”	无刷电机零点

举例：

BM , 60000	设定 B 轴无刷电机模数为 60000
BMC=100000/6	设定 C 轴无刷电机模数为 100000/6
BM , , , ?	查询 D 轴无刷电机模数

提示：更改 BM 参数会引起换向相位的立即改变。

22) BN

功能：烧录参数

说明：BN 命令将如下控制器的参数存储到控制器的 EEPROM 中，执行此命令需要 1 秒钟且不能中断。烧录完成后，控制器返回冒号（:）表示烧录成功。

烧录期间，BN 保存的参数：

AC	CO	GA	LZ	TL
AF	CW	GM	MO	TM
BA	DC	GR	MT	TR
BB	DV	IA	OE	VA
BI	EI	IL	OF	VD
BL	EO	IT	OP	VF
BM	ER	KD	PF	VS
BO	FA	KI	PL	VT
CE	FL	KP	SB	
CN	FV	KS	SP	

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	-

操作应用：

_BN 读取控制器的系列号

相关命令：

“BP”	烧录程序
“BV”	烧录变量

举例：

KD100	设定 A 轴微分常数
KP10	设定 A 轴比例常数
KI 1	设定 A 轴积分常数
AC200000	设定 A 轴加速度
DC150000	设定 A 轴减速度
SP10000	设定速度
MT -1	用“-1”设定 A 轴电机的类型为负向伺服电机

MO	关断电机
BN	烧录参数，需多达 15 秒时间

23) BO (二进制 9F)

功能：无刷电机偏置

说明：BO 命令在命令信号输出时设定一个偏置值，用于正弦波换向电机。用此对放大器设置任意偏置值，也可用于相位初始化设置。

参数：BO n,n,n,n,n,n,n 或 BOA=n

其中：n 指定电压值，为-9.998~+9.998 之间的符号数，其分辨率为 0.003。

n=? 返回指定轴的无刷电机偏置值。

用途：

运动进行中	No
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	0

操作应用：

_BO n 读取指定轴 DAC 输出端的偏置电压。

举例：

BO-2,, 1 对正弦波换向电机的 A 轴 DAC 端产生-2V 电压，C 轴的 DAC 端产生 1V 电压

相关命令：

“BA”	无刷电机轴
“BB (二进制 9E)”	无刷电机相位始点
“BC”	无刷电机通讯
“BD (二进制 9D)”	无刷电机相位角
“BI”	无刷电机输入
“BM (二进制 9B)”	无刷电机模数
“BS”	无刷电机设置
“BZ”	无刷电机零点

提示：为确保输出电压等于 BO 参数设定值，将 PID 和 OF 参数设置为 0。

24) BP

功能：烧录程序

说明：BP 命令将应用程序保存到控制器的非易失存储器 EEPROM 中。执行此命令一般需用 10 秒时间且不能中断，程序烧录完成后，返回一个冒号 (:)。

用途：

运动进行中	No
在程序中	No
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
-----	---

相关命令：

“BN”	烧录参数
“BV”	烧录变量

注：BP 命令使 GALIL 软件发出如下警告“A time-out occurred while waiting for a response from

the controller”。此警告是普遍的，当控制器在时间溢出期间内没有响应给命令，就会给用户发出警告。出现此警告是由于此命令需要多于 5 秒（缺省时间溢出值）时间。时间溢出能改变 GALIL 软件，但此警告不影响控制器或软件的操作。

25) BS

功能：无刷电机配置

说明：BS 命令设置正弦波换向无刷电机的配线。若连接霍尔传感器，此命令也测试霍尔传感器的配线，此功能一次只能执行一轴。

此命令返回与无刷电机配置相关的状态信息。下列信息将由控制器来返回：

- 无刷电机相位连线是否正确
- 无刷电机磁极周期的大约值
- BB 命令值（若使用霍尔传感器）
- 霍尔传感器连线测试结果（若使用霍尔传感器）

当执行完成，此命令将关断电机，而当电机关断时可以给定此命令。

一旦无刷电机配置正确，且保存在非易失性存储器中，就不需要重新执行 BS 命令。

用 BN 命令保存配置值。

注：为了正确进行无刷电机配置，必须至少让电机以两个方向运动一个磁极周期。

参数：BSA=v, n

其中：v 是 0~10 之间的实数，v 代表用于各相的电压值。

n 是 100~1000 的正整数，n 表示应该加到电机上的相位电压的幅值周期，用毫秒表示。

用途：

运动进行中	No
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	200

举例：

BSC=2, 900 对 C 轴进行配置测试，每次配置加入 900msec，2V 电压。

相关命令：

“BA”	无刷电机轴
“BB（二进制 9E）”	无刷电机相位始点
“BC”	无刷电机通讯
“BD（二进制 9D）”	无刷电机相位角
“BI”	无刷电机输入
“BM（二进制 9B）”	无刷电机模数
“BO”	无刷电机偏置
“BZ”	无刷电机零点

注：当使用 GALIL Windows 软件时，若执行 BS 命令，时钟超时必须设置成 10sec（Time out=10000），这样使软件能处理从控制器返回的所有信息。

26) BV

功能：将变量存入 FLASH

说明：BV 命令将变量与数组保存到控制器的非易失存储器（EEPROM），此命令需用 2 秒，且不能中断，烧录成功后返回一个冒号(;)。

参数：None

用途:	缺省:
运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

相关命令:

“BP” 将程序存入 FLASH

注: 此命令可使 GALIL 控制器发出如下警告 “A time-out occurred while waiting for a response from the controller”。此警告是普遍的，当控制器在时间溢出期间内没有响应给命令，就会给用户发出警告。出现此警告是由于此命令需要多于 1 秒（缺省时间溢出值）时间。时间溢出能改变 GALIL 软件，但此警告不影响控制器或软件的操作。

27) BZ

功能: 无刷电机零点

说明: BZ 命令用于正弦波换向配置的轴，此命令使电机转到磁极相位的零点，然后将换相相位设置为 0。
当电机关断时，可用此命令。

参数: BZ n,n,n,n,n,n,n 或 BZA= n 或 BZ<1

其中，n 为 -9.998~+9.998 之间的实数。参数 n 将设置初始化期间加到伺服放大器的电压。为准确起见，BZ 命令电压必须加大到足以使电机运动。如果参数为正数，当执行完 BZ 命令时，电机就会转入关掉状态（MO），如果参数为负值，就会使电机处于使能状态（SH）。

t 是 1~32762 之间的整数，代表 BZ 函数建立的时间。控制器将等待 t 秒更新采样周期（采样周期的缺省值为 1000usec）在零磁极相位时使电机稳定下来。在使用 BZ 命令之前应该定义 t 参数。

注: BZ 命令使电机短暂运动，推荐用小电压启动，然后逐渐增大。

注: 经常使用位置错误关断电机功能（OE），避免在测试正弦波换向时电机飞车。

用途:	缺省:
运动进行中	No
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省值 n=0, t=1000
缺省格式 0

操作应用:

_BZn 用编码器计数读取电机当前位置及指定轴换向零点位置的距离。此命令对于相位初始化，命令电机移到换向零点非常有用。

相关命令:

“BA”	无刷电机轴
“BB（二进制 9E）”	无刷电机相位始点
“BC”	无刷电机校准
“BD（二进制 9D）”	无刷电机相位角
“BI”	无刷电机输入
“BM（二进制 9B）”	无刷电机模数
“BO（二进制 9F）”	无刷电机偏置
“BS”	无刷电机安装

28) CA

功能：联动坐标轴

说明：CA 命令定义用于处理矢量的联动坐标系。以下命令适用于由 CA 命令设置的有效联动坐标系：

CR	ES	LE	LI	VM
TN	VE	VM	VP	

参数：CAS 或 CAT

其中：CAS 规定后续矢量命令将适用于 S 坐标系。

CAT 规定后续矢量命令将适用于 T 坐标系。

若 S 坐标系有效，CA? 返回值为 0，若 T 坐标系有效，CA? 返回值为 1。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	CAS
缺省格式	-

操作应用：

若 S 坐标系有效，_CA 读取值为 0，若 T 坐标系有效，_CA 读取值为 1。

相关命令：

“VP (二进制 B2)”	矢量位置
“VS (二进制 B9)”	矢量速度
“VD (二进制 B8)”	矢量减速度
“VA (二进制 B7)”	矢量加速度
“VM”	矢量方式
“VE”	矢量结束
“BG (二进制 A0)”	BGS_启动程序

举例：

CAT	定义 T 坐标系
VMAB	在 A 和 B 平面定义矢量运动 (XY 平面)
VS10000	定义矢量速度
CR1000, 0, 360	产生半径为 1000 计数单位的圆弧，从 0 度开始，以逆时针方向完成一个圆弧运动。
VE	矢量结束
BGT	启动 T 坐标系运动

29) CB (二进制 EB)

功能：清除位

说明：CB 命令将指定的输出位设置为低电平。也能够用 CB 来清除所配置的扩展 I/O 的输出位。

参数：CB n

其中：n 是与所要清除（设定为 0）的控制器的指定输出相关的整数。控制器第一个输出被视为输出点 1。

注意：当使用 Modbus 装置 (DMC-2100, DMC-2200) 时，用下式来计算 Modbus 装置的 I/O 点：

$$n = (\text{Slave Address} \times 10000) + (\text{Handle Num} \times 1000) + ((\text{Module}-1) \times 4) + (\text{BitNum}-1)$$

当 Modbus 装置有从装置与之相连时，使用 Slave Address（从地址），其地址范围为 0~255。请注意：对 Modbus 使用从地址非常少见，因此，此数值通常为 0。

Handle Num 是端口指定器，从 A~F。

Module 是模块在插槽中的位置，从 1~16。

BitNum 是模块的 I/O 点，从 1~4。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	-

相关命令：

“SB（二进制 EA）”	设置位
“OB（二进制 E9）”	输出位
“OP（二进制 E8）”	定义输出口（字节方式）

举例：

CB 7	清除输出位 7 为低电平
CB16	清除输出位 16 为低电平（只适用于 5~8 轴控制器）

30) CC

功能：配置通信口 2

说明：CC 命令为辅助串口配置波特率，握手传送方式和回声。在由通信口 2 使用 MG，IN 或 CI 命令之前，必须写入此命令。

参数：CC m, n, r, p

m—波特率	300, 1200, 4800, 9600, 19200, 38400
n—握手	0: 握手 OFF, 1: 握手 ON
r—方式	0: 菊花链 OFF, 1: 菊花链 ON
p—回声	0: 回声 OFF, 1: 回声 ON

用途：

运动进行中	Yes	缺省值	0, 0, 0
在程序中	Yes	缺省格式	-
命令行	Yes		
适用控制器	DMC-1200, 1600, 1700, 1800, 1842, 9542 之外		

缺省：

相关命令：

“CI”	中断通讯
------	------

举例：

CC 9600, 0, 0, 1	波特率：9600，握手 OFF，菊花链 OFF，回声 ON 用 TERM-P 或 TERM-H 时的典型设置
CC 19200, 1, 1, 0	波特率：19200，握手 ON，菊花链 ON，回声 OFF 用菊花链方式的典型设置

31) CD（二进制 BE）

功能：轮廓数据

说明：CD 命令定义 A, B, C, D 轴的增量位置。其值为编码器计数单位。此命令只使用于

轮廓方式（CM）。在由命令 DT（2~256 伺服更新周期）所定义的定时周期内执行增量位置。

参数： CD $n_1, n_2, n_3, n_4, n_5, n_6$ CDA= n

其中： n 是 ± 32762 之间的整数。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	-

相关命令：

“CM（二进制 BD）”	轮廓方式
“WC（二进制 D4）”	等待轮廓结束
“DT（二进制 BF）”	定时增量
“CS”	_CS 是段轮廓

举例：

CM ABCD	指定轮廓方式
DT 4	指定轮廓定时增量
CD 200, 350, -150, 500	指定 A, B, C, D 轴增量位置, A 轴运动 200 计数单位, B 轴运动 350 计数单位, C 轴运动 -150 计数单位, D 轴运动 500 计数单位。
WC	等待轮廓方式
CD 100, 200, 300, 400	新的位置数据
WC	等待轮廓完成
DT0	停止轮廓运动
CD 0, 0, 0, 0	退出轮廓运动方式

32) CE（二进制 8C）

功能： 配置编码器

说明： CE 命令将编码器配置成差分四倍频型或脉冲+方向型。也允许为了调换反馈方向而转换编码器的极性。

注意：当使用伺服电机时，若在负反馈情况下，只转换编码器极性，电机不会暴走（飞车）。此项配置可分别运用于主编码器和辅助编码器。

参数： CE $n_1, n_2, n_3, n_4, n_5, n_6$ 或 CEA= n

其中： n 是 0~15 之间的整数。每个整数是 M 和 N 两个整数之和，M 为主编码器类型，N 为辅助编码器类型。M 和 N 的值如下：

M=	主编码器类型	N=	辅助编码器类型
0	通用差分四倍频型	0	通用差分四倍频型
1	通用脉冲+方向型	4	通用脉冲+方向型
2	反向差分四倍频型	8	反向差分四倍频型
3	反向脉冲+方向型	12	反向脉冲+方向型

例如： $n=10$ 意味着 $M=2$, $N=8$ ，即两个编码器均为反向差分四倍频型。

$n=?$ 返回指定轴编码器的配置值。

用途：

缺省：

运动进行中	Yes	缺省值	0
在程序中	Yes	缺省格式	2.0
命令行	Yes		
适用控制器	所有型号控制器		

操作应用：

_CEn 读取该轴由“n”指定的编码器类型值。

相关命令：

“MT” 定义电机类型

举例：

CE 0, 3, 6, 2	配置编码器
CE ?, ?, ?, ?	查询配置值
V=_CEA	分配配置为一个变量

注意：当使用脉冲+方向编码器时，脉冲信号连接到 CHA，方向信号连接到 CHB。

33) CF

功能：配置

说明：为信息传送设置缺省通信口。通过缺省配置，若 ENET/USB 指拨开关为 OFF，DMC-2x00 会主动发送响应到主 RS-232 串口，否则，DMC-2x00 就会发送这些响应到上次给控制器发送信息的那个 Ethenet 端口。CF 命令允许用户调整 ENET/USB 指拨开关，以便发送响应到主串口、辅助串口或端口 A~F。

参数：CF n

其中：对于 DMC-2100, 2200: n 是 A~F，对应 Ethenet 端口 1~6，S 对应主串口，T 对应辅串口，N 读取 ENET 开关状态。

对于 DMC-2000: 主串口时，n 为 S，辅串口时，n 为 T，对于 USB，n 为任意其它字母，N 读取 USB 开关状态。

用途：

缺省：

运动进行中	Yes	缺省值	-
在程序中	Yes	缺省格式	-
命令行	No		
适用控制器	DMC-2000, 2100, 2200		

34) CI

功能：中断通信

说明：CI 命令配置以串口 2，辅助串口所接收到的字符为基础的程序中断。中断使程序流程跳转到#COMINT 子程序。如果使用多个程序域，#COMINT 子程序在域 0 中运行，而域 1, 2, 3 在无中断的后台运行。由串口接收到的字符保存在内部变量中，如 P2CH。有关通信中断方面更为详细的说明，参见用户手册第 7 章。

参数：CIn

参数	解释
n=0	不中断串口 2
n=1	敲回车时，串口 2 字符中断
n=2	由串口 2 任意字符中断
n=-1	清除中断数据缓存

用途：

缺省：

运动进行中	Yes	缺省值	-
在程序中	Yes	缺省格式	-
命令行	No		
适用控制器	DMC-2000, 2100, 2200		

相关命令:

“CC”	设定 Slave 通讯端口
“IN”	输入变量
“MG”	显示信息

举例:

CI 1	当串口 2 上接收到<ENTER>回车键符时中断
CI 2	由串口 2 接收到单一字符中断。

35) CM (二进制 BD)

功能: 轮廓方式

说明: 轮廓方式由指令 CM 来定义。此方式让用户产生任意轴之间的任意运动包络线。CD 命令指定位置增量, DT 命令指定时间间隔。能够用 CM? 命令来检查轮廓缓存状态。命令 CM? 返回的值为 1 表示轮廓缓存处于满。返回值为 0 则表示轮廓缓存为空。

参数: CM nnnnnnnn

其中: n 是 A, B, C, D, E, F, G 或为轮廓方式指定坐标轴的任意组合。

n=? 若轮廓缓存为满, 返回值为 1, 若轮廓缓存为空, 则为 0。

用途:

运动进行中	Yes	缺省值	0
在程序中	Yes	缺省格式	2.0
命令行	Yes		
适用控制器	所有型号控制器		

缺省:

操作应用:

_CM 若轮廓缓存为空, 读取值为 0, 否则为 1

相关命令:

“CD (二进制 BE)”	轮廓数据
“WC (二进制 D4)”	等待轮廓完成
“DT (二进制 BF)”	定时增量

举例:

V=_CM; V=	返回轮廓缓存状态
CM?	返回轮廓缓存状态
CM AC	指定 A, C 轴为轮廓方式

36) CN (二进制 E6)

功能: 配置

说明: CN 命令配置限位开关、原点返回开关、锁存输入和选择性急停输入点的极性。

参数: CN m, n, o, p

其中: m, n, o 是 1 或-1 整数, p 是 0 或 1 整数。

m=	1	限位开关输入高电平有效
	-1	限位开关输入低电平有效

n=	1	当此输入为高电平时，电机以正向运动逼近原点开关，参见 HM 和 FE 命令
	-1	当此输入为高电平时，电机以反向运动逼近原点开关，参见 HM 和 FE 命令
o=	1	锁存输入为高电平有效
	-1	锁存输入为低电平有效
p=	1	分别配置输入点 5、6、7、8、13、14、15、16 作为 A、B、C、D、E、F、G、H 轴的选择急停输入
	0	配置输入点 5、6、7、8、13、14、15、16 为通用输入点

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-1, -1, -1, 0
缺省格式	2.0

操作应用：

_CN0	读取限位开关配置值
_CN1	读取原点开关配置值
_CN2	读取锁存输入配置值
_CN3	读取选择性急停功能的状态（1：使能，0：无效）

相关命令：

“AL（二进制 EE）” 锁存使能

举例：

CN 1, 1	设置限位、原点开关为高电平有效
CN , , -1	设置输入锁存为低电平有效

37) CO

功能：配置输出

说明：CO 命令为 DMC-1600, 17x8 和 DMC-2x00 配置扩展 I/O。

注意：对于 DMC1700 系列控制器来说，此命令只适用于 DMC-17x8，可以对 DMC-17x8 系列控制器的 64 点扩展 I/O 以 8 块加以配置。扩展 I/O 表示为 Bit17~80 和块 2~9。

参数：CO_n

其中：n 是代表二进制数的十进制值。二进制数的各位代表扩展 I/O 的一个块。当设定为 1 时，相应的块就被配置为输出。最低有效位代表块 2，而最高有效位代表块 9。由下式来计算十进制值： $n = n_2 + 2 \times n_3 + 4 \times n_4 + 8 \times n_5 + 16 \times n_6 + 32 \times n_7 + 64 \times n_8 + 128 \times n_9$ 。其中： n_x 代表块。要想将块配置成输出块，就将 1 代入方程式中的 n_x 。如果 n_x 为 0，8 个 I/O 点的块将被配置为输入。例如：若要将块 3、4 配置为输出。就为 CO6

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	DMC-1600, 1700*, 1800*, 17 x8, 2000, 2100, 2200

缺省：

缺省值	-
缺省格式	-

* 带有扩展 I/O 板的 DMC-1700, 1800

操作应用：

_CO	读取输出配置值
-----	---------

相关命令:

“CB (二进制 EB)”	清除输出位
“SB (二进制 EA)”	设置输出位
“OP (二进制 E8)”	设置输出口
“TI (二进制 E0)”	响应输入点

举例:

CO 0	配置所有点为输入点
CO 1	配置扩展 I/O 的块 1 为输出

提示: 有关扩展 I/O 板的详细信息, 参见附录。

38) CR (二进制 B3)

功能: 圆弧

说明: CR 命令规定二维圆弧线段, 半径为 r , 起始角为 θ , 夹角为 $\Delta\theta$ 。正向夹角 $\Delta\theta$ 表示逆时针方向运动, 负向夹角 $\Delta\theta$ 表示顺时针方向运动。在所有 CR 和 VP 线段规定之后, 必须使用 VE 命令来定义运动次序结束。用 BG 命令来启动运动时序。必须指定 r , θ , $\Delta\theta$ 所有参数。半径单位为四倍频计数单位。 θ 和 $\Delta\theta$ 为角度单位。参数 n 是可选的, 它定义与运动线段相关的矢量速度。

参数: CR r , θ , $\Delta\theta$ $< n > 0$

其中: r 是无符号实数, 范围为 10~6000000 十进制数 (半径)

θ 是带符号数, 范围为 0~ ± 32000 十进制数 (起始角度)

$\Delta\theta$ 是带符号实数, 范围为 0.0001~ ± 32000 十进制数 (角度)

n 指定执行矢量线段运动时的矢量速度。对于伺服电机来说, n 是无符号偶整数, 范围为 0~12000000, 对于步进电机来说, 范围为 0~3000000。

O 指定要到达矢量线段终点的矢量速度。O 是无符号偶整数, 范围为 0~8000000。

注意: $r \times \Delta\theta$ 的乘积必须限定在 $\pm 4.5 \times 10^8$ 范围内。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	-
缺省格式	-

相关命令:

“VP (二进制 B2)”	矢量位置
“VS (二进制 B9)”	矢量速度
“VD (二进制 B8)”	矢量减速度
“VA (二进制 B7)”	矢量加速度
“VM”	矢量方式
“VE”	矢量结束
“BG (二进制 A0)”	BGS一开始顺序运动

举例:

VMAB	在 A, B 平面指定矢量运动
VS 10000	指定矢量速度
CR 1000, 0, 360	产生半径为 1000 计数单位, 起始角为 0 度, 夹角为 360 度的圆弧运动。
CR 1000, 0, 360<40000	产生半径为 1000 计数单位, 起始角为 0 度, 夹角为

VE	360 度的圆弧运动。
BGS	结束矢量程序 开始矢量运动

39) CS

功能: 清除矢量程序

说明: CS 将会对 S 或 T 坐标系驻留在运动程序中的 VP, CR, LI 命令进行清除。在执行完程序之后, 就没有必要将 CS 命令放入一个新程序中。当您错误地指定了 VP,, CR, LI 命令时, 此命令非常有用。

参数: CSS 或 CST

其中: 可以用参数 S 或 T 清除 “S” 或 “T” 坐标系程序缓存区。

用途:

运动进行中	No
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	-
缺省格式	-

操作应用:

_CSn 读取 S、T 坐标系中, 由 n 指定的程序线段数。此操作数在直线插补方式 LM、矢量方式 VM 中有效。

相关命令:

“CR (二进制 B3)”	圆弧插补线段
“LI (二进制 B1)”	直线插补距离
“LM (二进制 B0)”	直线插补方式
“VM”	矢量方式
“VP (二进制 B2)”	矢量位置

举例:

#CLEAR	标号
CAT	指定 T 坐标系矢量点
VP 1000, 2000	矢量位置
VP 4000, 8000	矢量位置
CST	清除在 T 坐标系中指定的矢量
CAS	指定 T 坐标系矢量点
VP 1000, 5000	新矢量
VP 8000, 9000	新矢量
CSS	清除在 S 坐标系中指定的矢量

40) CW

功能: 版权信息/数据调整位 ON/OFF

说明: CN 命令有两个用途。当参数 n=0 时, CW 命令读取版权信息, 否则, 由伺服参数调整软件 WSDK 用作为通信增强之用。当是 ON 时, 通信增强设置未经请求的 MSB, 返回的 ASCII 字符为 1, 未经请求的 ASCII 字符是那些不用直接从终端询问而从控制器返回的字符。这是出现在程序有需要控制器返回数值或字符中的命令时, 由于个有两个功能, 一次只能设置一种用途。用 CW 2; CW , 1, 取代 CW2, 1。

参数: CW n, m

其中：n=0 使控制器返回版权信息
 n=1 使控制器设置返回字符未经请求的 MSB 为 1
 n=2 使控制器不设置未经请求的 MSB
 n=? 读取控制器版权信息
 m 为可选取参数
 m=0 当输出 FIFO 满时，控制器暂停程序执行，当 FIFO 不满时，恢复程序执行。

用途：		缺省：	
运动进行中	Yes	缺省值	2, 0
在程序中	Yes	缺省格式	----
命令行	Yes		
适用控制器	所有型号控制器		

操作应用：

_CW 读取数据调整位的值。2=OFF, 1=ON。

注意： CW 命令会对控制器返回的字符产生篡改。控制器的缺省状态为 CW 命令无效，不过，为便于内部使用，GALIL WSDK 和终端软件时常对 CW 命令使能。如果在 GALIL 软件运行期间使控制器复位，可能会使 CW 命令复位到缺省值，这会对软件运行带来麻烦。此时，有必要对 CW 命令重新使能。CW 命令状态可存储在 EEPROM 中。

41) DA

功能： 释放变量及数组大小

说明： DA 命令释放数组和变量存储器空间。在此命令中，能够为存储器释放一个以上的数组或变量。当在一条命令中定义不同的数组和变量，需用逗号分开，参数 * 释放所有变量；*[0]释放所有数组。

参数： DA C[0]，变量名

其中：C[0]=所定义的数组名

变量名=所定义的变量名

*-释放所有变量

*[0]-释放所有数组

DA? 返回控制器中可用的数组数量。

用途：		缺省：	
运动进行中	Yes	缺省值	----
在程序中	Yes	缺省格式	----
命令行	Yes		
适用控制器	所有型号控制器		

操作应用：

_DA 读取可用变量的总数。例如，在定义变量之前，读取_DA 为 30，如果定义了一个变量，读取_DA 将返回 29。

相关命令：

“DM” 定义数组

举例： ‘Cars’ 和 ‘Sales’ 均为数组，且 ‘Total’ 是变量

DM Cars[400], Sales[500] 定义 2 个数组

Total=70 分配 70 给变量 Total

DA Cars[0], Sales[0], Total 释放 2 个数组和变量

DA *[] 释放所有数组

DA *, *[]

释放所有数组和变量

注意：此命令释放空间并压缩存储器中的数组空间，执行此命令可能占用 2ms 以上的时间。

42) DC (二进制 91)

功能：减速度

说明：减速度命令 DC 为独立运动如 PR, PA, JG 运动设定电机线性减速度。参数以 1024 的倍数取整，单位为计数单位/sec²。

参数：DC n, n, n, n, n, n, n, n 或 DCA=n

其中：n 是无符号数，范围为 1024 ~67107840

n=? 读取指定轴减速度值

用途：

运动进行中 Yes
在程序中 Yes
命令行 Yes

适用控制器 所有型号控制器

缺省：

缺省值 256000
缺省格式 8.0

* 在运动时，DC 命令只能针对 JOG 方式进行指定。

操作应用：

_DCn 读取指定轴的减速度。

相关命令：

“AC (二进制 90)”	加速度
“PR (二进制 A7)”	相对位置
“PA (二进制 A6)”	绝对位置
“SP (二进制 EB)”	速度
“JG (二进制 A8)”	JOG

举例：

PR 1000	指定位置
AC 200000	指定加速度
DC 100000	指定减速度
SP 300	指定速度
BG	开始运动

注意：JG 方式运动期间，可以改变 DC 命令值，但 PR 或 PA 方式中，在运动期间不能改变。

43) DE (二进制 98)

功能：辅编码器位置

说明：DE 命令定义辅编码器的位置。

当使用步进电机时，DE 命令定义编码器位置。

注意：辅编码器不适用于步进电机轴或输出比较处于有效的任何轴。

参数：DE nnnnnnnn 或 DEA=n

其中：n 是带符号整数，范围为-2147483647 ~2147483648 之间的十进制数

n=? 返回指定轴的辅编码器的位置反馈值。

n=? 当使用步进电机时，返回电机步进脉冲的命令参考值。例如：DE0，此命令将 TP 或编码器位置反馈值定义为 0。它不会影响 DE? 值。

(当在步进方式使用 DP 命令时，要设置 DE 值。)

用途：

缺省：

运动进行中	Yes	缺省值	0, 0, 0, 0
在程序中	Yes	缺省格式	位置格式
命令行	Yes		
适用控制器	所有型号控制器		

操作应用：

_DEn 读取指定辅编码器的当前位置。

相关命令：

“DP（二进制 97）”	定义主编码器位置
“TD（二进制 DB）”	响应辅编码器位置

举例：

DE 0, 100, 200, 400	设置 A, B, C, D 轴辅编码器当前位置为 0, 100, 200, 400
DE ?, ?, ?, ?	返回辅编码器位置
DUALA=_DEA	分配 A 轴辅编码器位置为变量 DUALA

提示：在电机和负载侧均需要编码器的场合，双编码器反馈将非常有用。负载侧的编码器通常为辅编码器，用此来校准负载侧实际位置。以负载侧位置误差来修正电机位置。

44) DL

功能：下载

说明：DL 命令将数据文件从主计算机传送到控制器。文件中的指令以无行号的数据流接收。

用<Control>Z、<Control>Q、<Control>D 等终止文件。在各条命令之前不要插入空格。如果没有指定参数，下载一个数据文件将清除控制器 RAM 中的所有程序。数据进入以 0 行开始保存。若没有太多行或每行没太多字符，控制器将会返回一个？。要想下载一个程序在标号后，则跟随 DL 指定标号名。参数#可与 DL 一起使用来注释文件，位于 RAM 中程序的最后。

使用 GALIL DOS 终端软件：ED 命令使控制器进入 Edit 子系统。在 Edit 子系统中，就能创建、修改、删除程序。Edit 子系统下的命令如下：

<Control>D	删除一行
<Control>I	在当前行之前插入一行
<Control>P	显示前行
<Control>Q	退出 Edit 子系统
<Return>	保存一行

参数：DLn

其中：n=无参数	从 0 行开始下载程序，删除 RAM 中的所有程序
n=#label	在跟随有#lable 的行开始下载
n=#	在 RAM 中的程序末端开始下载

用途：

缺省：

运动进行中	Yes	缺省值	----
在程序中	No	缺省格式	----
命令行	Yes		
适用控制器	所有型号控制器		

操作应用：

当用作操作数时，_DL 给定可用标号的数量。

所有 Optima 系列控制器有 254 个可用标号

相关命令：

“UL” 上载

举例：

DL;	开始下载
#A; PR4000; BGA	数据
AMA; MG DONE	数据
EN	数据
<Control>Z	结束下载

45) DM**功能：**尺寸**说明：**DM 命令用数组中的名称和元素数量定义单个数组大小。所定义数组的第一个元素以 0 号元素开始，最后一个元素为 n-1 号。**参数：**DMC[n]

其中：C 是多于 8 个字符的名称，以大写阿拉伯字母开始。n 指定数组的大小（数组元素的数量）。

n=? 返回数组元素可用数量

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	----
缺省格式	----

操作应用：

_DM 读取可用数组空间，例如：在定义数组之前，操作数_DM 将会返回 8000。如果定义 100 个元素的数组，_DM 将返回 7900

相关命令：

“DA” 释放数组

举例：

DM Pets[5], Dogs[2], Cats[3]	定义数组大小，Pets 为 5 个元素，Dogs 为 2 个元素，Cats 为 3 个元素。
DM Test[1600]	定义数组 Tests 为 1600 个元素

46) DP (二进制 97)**功能：**定义位置**说明：**DP 命令将当前电机位置和当前命令位置设置成用户所指定的值，比值为 4 倍频计数单位。此命令将设置 TP 和 RP 值。

对于配置为步进电机轴而言，DP 命令设置命令参考位置，单位为步。

例如：DP0，会将 TD 和 RP 寄存器设置为 0，但不影响 TP 寄存器值。

参数：DP n, n, n, n, n, n, n, n 或 DPA=n

其中：n 是带符号整数，范围为-2147483648~2147483647 之间的十进制数。

n=? 返回指定轴电机的当前位置。

用途：

运动进行中	No
在程序中	Yes

缺省：

缺省值	0, 0, 0, 0
缺省格式	位置格式

命令行	Yes
适用控制器	所有型号控制器

操作应用:

_DPn 读取指定轴的当前位置。

相关命令:

“PF”	位置格式
------	------

举例:

DP 0, 100, 200, 400	设置 A 轴当前位置为 0, B 轴为 100, C 轴为 200, D 轴为 400
DP , -5000	设置 B 轴当前位置为-50000, A, C, D 轴保持不变
DP ? , ? , ? , ?	查询 A, B, C, D 轴当前位置
0000000, -50000, 0000200, 0000400	返回所有 4 个轴电机位置
DP?	查询 A 轴位置
0000000	返回 A 轴位置

提示: DP 命令对于重新定义绝对位置非常有用。例如, 用电机关断命令 MO 使控制器使能断开, 用手使电机转动, 用 SH 命令使伺服电机使能, 然后用 DP0 重新将新位置定义为您的绝对零点。

47) DR

功能: 配置次通信通道和数据更新速率

说明: DR+/-指定次通信通道。若为正, 指定为 DMA 方式, 若为负, 指定为 FIFO 方式。n 指定数据更新速率为 2^n 采样速率。控制器以此速率创建一个记录, 并将其放在 FIFO 中或 PC DMA 中。

参数: DR +n 或 DR-n
+DMA (仅 DMC-1700)
-FIFO

其中: n 是 0~8 之间的整数。当 n=0 时, 关闭次通信通道, 当 n=1~8 指定 2^n 采样周期为数据更新速率。采样周期用 TM 命令来指定, 其缺省值 1000 (1msec)。

注意: 如果使用小采样周期和小更新速率, 控制器中的处理器有可能由于要维持高的更新速率而明显变慢。

用途:

运动进行中	Yes
在程序中	No
命令行	Yes
适用控制器	除 DMC-1200, 2000, 2100, 2200 之外

缺省:

缺省值	0
缺省格式	----

相关命令:

“QZQZ”	设定数据格式
--------	--------

48) DT (二进制 BF)

功能: 定时间隔

说明: DT 命令为轮廓方式设置定时间隔。发送一次 DT 命令将为所有轮廓数据设置定时间隔, 除非发送一个新 DT 命令。定时间隔为 2^n msec (由 CD0 命令跟随)

参数: DTn

其中：n 是 0~8 之间的整数。

n=0 终止轮廓方式。

n=1~8 指定 2^n 采样周期的定时间隔。

采样周期的缺省值为 1msec（由 TM 命令设定）；n=1，意味着定时间隔为 2msec。

n=? 返回轮廓方式定时间隔值

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	1.0

操作应用：

_DT 读取轮廓方式定时间隔值。

相关命令：

“CM（二进制 BD）”	轮廓方式
“CD（二进制 BE）”	轮廓数据
“WC（二进制 D4）”	等待下一个数据

举例：

DT 4	指定定时间隔为 16msec
DT 7	指定定时间隔为 128msec
#CONTOUR	开始标号
CMAB	进入轮廓方式
DT4	设定定时间隔
CD 1000, 2000	指定数据
WC	等待轮廓完成
CD 2000, 4000	新数据
WC	等待轮廓完成
DT0	停止轮廓
CD0	退出轮廓方式
EN	程序结束

49) DV（二进制 84）

功能：双速度环（双环）

说明：DV 函数改变滤波器的运作方式。它使 KD（微分）命令在双编码器上操作，而不是只为主编码器上操作。若在电机和主编码器之间有间隙，且安装编码器的情况下，会提高系统稳定性。

参数：DV n, n, n, n, n, n, n, n 或 DVX=n

其中：n=0 双环方式无效。

n=1 双环方式使能

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	----

操作应用：

_DVn 读取指定轴的双速度方式的状态，0=无效，1=使能。

相关命令：

“KD (二进制 83)”	微分常数
“FV (二进制 95)”	速度前馈

举例：

DV 1, 1, 1, 1	对所有轴双环方式使能
DV 0	A 轴双环无效
DV ,, 1, 1	对 C、D 轴双环使能，其它轴保持不变
DV 1, 0, 1, 0	对 A、C 轴双环使能，B、D 轴双环无效
MG _DVA	读取 A 轴双环方式的状态

提示： DV 命令对间隙和滞后补偿很有用。

50) EA

功能： 选择 ECAM 主动轴

说明： EA 命令为电子凸轮方式选择主动轴。可以选择任意一轴为主动轴。

参数： EA n

其中：n 是 A、B、C、D、E、F、G、H 或 N 中的其中之一。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	----
缺省格式	-----

相关命令：

“EB (二进制 C4)”	ECAM 使能
“EC (二进制 C6)”	设置 ECAM 表标志
“EG (二进制 C3)”	使 ECAM 生效
“EM (二进制 C1)”	指定 ECAM 周期
“EP (二进制 C2)”	指定 ECAM 表间隔和起始点
“EQ (二进制 C5)”	退出 ECAM
“ET (二进制 C0)”	ECAM 表

举例：

EAB 选择 B 轴为 ECAM 主动轴

51) EB (二进制 C4)

功能： 使 ECAM 有效

说明： EB 函数使电子凸轮方式有效或无效。在此方式中，主动轴的起始位置被规定在一个周期之内，当给定 EB 命令时，主动轴就被激活。

参数： EB n

其中：n=1 开始 ECAM 方式
n=0 停止 ECAM 方式
n=? 若 ECAM 无效，n=0，若 ECAM 有效，返回值为 1

用途：

运动进行中	Yes
在程序中	Yes

缺省：

缺省值	0
缺省格式	1.0

命令行	Yes
适用控制器	所有型号控制器

操作应用：

EB 读取 ECAM 方式当前状态。0=无效，1=使能。

相关命令：

“EA”	选择 ECAM 主动轴
“EC (二进制 C6)”	设置 ECAM 表标志
“EG (二进制 C3)”	ECAM 生效
“EM (二进制 C1)”	指定 ECAM 周期
“EP (二进制 C2)”	指定 ECAM 表间隔和起始点
“EQ (二进制 C5)”	ECAM 停止
“ET (二进制 C0)”	ECAM 表

举例：

EB1	开始 ECAM 方式
EB0	停止 ECAM 方式
B=_EB	返回 CAM 方式的状态

52) EC (二进制 C6)

功能：ECAM 计数

说明：EC 函数将标志设定到 ECAM 表中。此命令只用于没有标志值而进入 ECAM 表时，当以二进制发送命令时最适用。详见命令 ET。

参数：EC n

其中：n 是 0~256 之间的整数

n=? 返回进入 ECAM 表的当前标志值

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	1.0

操作应用：

EC 读取进入 ECAM 表的当前标志值。

相关命令：

“EA”	选择 ECAM 主动轴
“EB (二进制 C4)”	使 ECAM 有效
“EG (二进制 C3)”	ECAM 生效
“EM (二进制 C1)”	指定 ECAM 周期
“EP (二进制 C2)”	指定 ECAM 表间隔和起始点
“EQ (二进制 C5)”	ECAM 停止
“ET (二进制 C0)”	ECAM 表

举例：

EC0	设置 ECAM 标志为 0
ET 200, 400	设置第一个 ECAM 表入口为 200, 400
ET 400, 800	设置第二个 ECAM 表入口为 400, 800

53) ED

功能：编辑

说明：使用 **GALIL DOS 终端软件**：ED 命令使控制器进入编辑子系统，在编辑子系统中，能够创建、更改、删除程序，编辑子系统中的命令如下：

<Cntrl>D	删除一行
<Cntrl>I	在当前行之前插入一行
<Cntrl>P	显示前面一行
<Cntrl>Q	退出编辑子系统
<Return>	保存一行

使用 GALIL Windows 终端软件：ED 命令使 Windows 终端软件开启终端编辑器。

操作应用：

_ED	读取出错的上一行的行号
_ED1	读取出错的域数（用于多任务）

举例：

```
ED
000 #START
001 PR 2000
002 BGA
003 SLKJ                                不正确行
004 EN
005 #CMDERR                            错误查询子程序
006 V=_ED
007 MG "An error has occurred" {n}
008 MG "In line", V{F3.0}
009 ST
010 ZS0
011 EN
XQ _ED2                                重试含有错误的指令
XQ _ED3                                执行下一条指令
```

提示：在执行程序或对程序列表之前，请勿忘退出编辑方式。

54) EG (二进制 C3)

功能：ECAM 执行

说明：EG 命令会在主动轴的指定位置启动 ECAM 从动轴，如果所指定的起始位置值在主动轴的范围外侧，从动轴就会立即启动。一旦启动从动轴，就重新定义从动轴位置以使其在凸轮周期内。

参数：EG n, n, n, n, n, n, n, n 或 EGA=n

其中：n 是 ECAM 主动轴位置，必须在此位置启动 ECAM 从动轴。

n=? 如果所指定轴启动，返回值为 1，否则为 0。

适用控制器 所有型号控制器

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	1.0

操作应用：

_EGn 读取指定轴的 ECAM 状态。0=轴未启动，1=轴被启动。

相关命令：

“EA”	选择 ECAM 主动轴
“EB（二进制 C4）”	使 ECAM 有效
“EC（二进制 C6）”	设置 ECAM 表标志
“EM（二进制 C1）”	指定 ECAM 周期
“EP（二进制 C2）”	指定 ECAM 表间隔和起始点
“EQ（二进制 C5）”	ECAM 停止
“ET（二进制 C0）”	ECAM 表

举例：

EG 700, 1300 分别在主动轴位置 700 和 1300 处启动 A 轴和 B 轴

B=_EGB 读取 B 轴状态，若被启动，为 1

注意：此命令不是条件启动命令。此命令不会使程序流继续执行。如果需要使程序继续执行直至到达主动轴位置，请使用 MF 或 MR 命令。

55) EI（二进制 ED）

功能：中断使能

说明：EI 命令中断条件使能，如运动完成或运行报警等。用参数 m 选择中断条件，其中 m 是所选条件的位屏数，如下所示。在使用中断之前，必须先配置中断，并在控制器程序中事先配备中断服务程序。

参数：EI m, n

其中：EI0 清除中断 que

m 是 1~65535 之间的整数。用来选择要使用的中断。

n 是 1~255 之间的整数。若 m 值代表用于中断的输入，就用 n 来选择所指定的输入。

m 和 n 是表示二进制数的整数值。这些二进制数与中断寄存器进行逻辑与，如下表所示。利用 m 和 n 从中断寄存器选择想要的位称之为‘屏蔽’。

位号	m=2 ^{位号}	条件	位号	m=2 ^{位号}	条件
0	1	A 轴运动完成	8	256	所有轴运动完成
1	2	B 轴运动完成	9	512	位置超差*
2	4	C 轴运动完成	10	1024	限位开关
3	8	D 轴运动完成	11	2048	Watchdog 定时
4	16	E 轴运动完成	12	4096	保留
5	32	F 轴运动完成	13	8192	应用程序被停止
6	64	G 轴运动完成	14	16384	命令执行完
7	128	H 轴运动完成	15	32768	输入*（用 n 来屏蔽）

* 在每次出现之后，必须重新使能

位号	m=2 ^{位号}	条件	位号	m=2 ^{位号}	条件
0	1	Input1	4	16	Input5
1	2	Input2	5	32	Input6
2	4	Input3	6	64	Input7
3	8	Input4	7	128	Input8

注意：如果用 Bit15 对中断功能使能，只使用 n 即可。

用途：		缺省：	
运动进行中	Yes	缺省值	0
在程序中	Yes	缺省格式	----
命令行	Yes		
适用控制器	DMC-2000, 2100, 2200 除外		

相关命令：

“UI” 用户中断

举例：

- a. 指定所有轴运动完成时的中断与限位发生的中断相或；从表中得知，Bit8 和 Bit10 使能；所以 $m=2^8+2^{10}=256+1024=1280$ EI 1280
- b. 指定 Input3 中断
 m 中的 Bit15 和 n 中的 Bit2 使能。
 所以： $m=2^{15}=32768$
 $n=2^2=4$
 EI 32768, 4

56) ELSE

功能：ELSE 函数与 IF 条件语句一起使用。

说明：ELSE 命令是 IF 条件语句的选择部分。ELSE 命令必须出现在 IF 命令之后，且没有参数。只有在 IF 命令的参数不具备时才执行 ELSE 命令。如果 IF 命令的参数不具备，控制器就会跳过一些命令而到达 ELSE 命令。如果 IF 命令的参数为真，控制器就执行 IF 和 ELSE 命令之间的那些命令。

参数：ELSE

用途：		缺省：	
运动进行中	Yes	缺省值	
在程序中	Yes	缺省格式	
命令行	Yes		
适用控制器	所有型号控制器		

相关命令：

“ENDIF” IF 条件语句的结束

举例：

IF (@IN[1]=0)	以 Input 1 为基础的 IF 条件语句
IF (@IN[2]=0)	若第一条 IF 条件为真，执行第 2 条 IF 条件语句
MG “Input 1 and input 2 are active”	若第二条 IF 条件为真，发送信息
ELSE	用于第二条 IF 条件语句的 ELSE 命令
MG “Only input 1 is active”	若第二条 IF 条件为假，发送信息
ENDIF	第二个条件语句的结尾
ELSE	第一个 IF 条件语句的 ELSE 命令
MG “Only input 2 is active”	若第一个 IF 条件为假，发送信息
ENDIF	第一个条件语句的结尾

57) EM (二进制 C1)

功能：凸轮周期



说明: EM 命令属于 ECAM 方式部分。用它来定义主动轴一个完整周期过后位置变更。主动轴的区域是运动轴位置的循环。对于从动轴来说，区域定义一个周期中改变部分，如果从动轴返回到循环终点处的初始位置，此变更值为零。若变更为负值，就以绝对值来指定。

参数: EM n, n, n, n, n, n, n, n, n 或 EMA=n

其中,对于主动轴,n是1~8388607之间的正整数,对于从动轴,其范围为1~2147483647。

用途:

运动进行中 Yes

在程序中 Yes

命令行 Yes

适用控制器	所有型号控制器
<p>1. 在“主菜单”中，选择“系统设置”。</p> <p>2. 在“系统设置”中，选择“安全设置”。</p> <p>3. 在“安全设置”中，选择“安全设置”。</p> <p>4. 在“安全设置”中，选择“安全设置”。</p> <p>5. 在“安全设置”中，选择“安全设置”。</p> <p>6. 在“安全设置”中，选择“安全设置”。</p> <p>7. 在“安全设置”中，选择“安全设置”。</p> <p>8. 在“安全设置”中，选择“安全设置”。</p> <p>9. 在“安全设置”中，选择“安全设置”。</p> <p>10. 在“安全设置”中，选择“安全设置”。</p>	<p>1. 在“主菜单”中，选择“系统设置”。</p> <p>2. 在“系统设置”中，选择“安全设置”。</p> <p>3. 在“安全设置”中，选择“安全设置”。</p> <p>4. 在“安全设置”中，选择“安全设置”。</p> <p>5. 在“安全设置”中，选择“安全设置”。</p> <p>6. 在“安全设置”中，选择“安全设置”。</p> <p>7. 在“安全设置”中，选择“安全设置”。</p> <p>8. 在“安全设置”中，选择“安全设置”。</p> <p>9. 在“安全设置”中，选择“安全设置”。</p> <p>10. 在“安全设置”中，选择“安全设置”。</p>

缺省:

缺省值

缺省格式

操作应用:

EMn	读取指定轴的周期
-----	----------

相关命令:

“EA” 选择 ECAM 主动轴

“EB（二进制 C4）” 使 ECAM 有效

“EC (二进制 C6)” 设置 ECAM 表标志

“EG (二进制 C3)”	ECAM 执行
---------------	---------

“EP（二进制 C2）” 指定 ECAM 表间隔和起始点

“EO (二进制 C5)” ECAM 停止

“ET (二进制 C0)” ECAM 表

举例:

EAC 选择 C 轴为 ECAM 的主动轴

EM 0, 3000, 2000 定义 A 轴和 B 轴的变更值分别为 0 和 3000。

定义主动轴周期为 2000。

V= EMA 返回 A 轴周期值

58) EN

功能：结束

说明：用 EN 命令来表明程序或子程序结束。如果一个子程序由 JS 命令来调用，EN 命令就使子程序结束，并使程序流返回到 JS 命令之后的那条指令。

用 EN 命令来结束自动子程序#MCTIME、#CMDERR 和#COMINT。当用 EN 命令来终止#COMINT 通信中断子程序时，有两个参数，子程序一旦完成，就首先决定是否存储条件启动，并重新决定是否对通信中断重新使能。

参数: EN m, n

其中：m=0：不存储条件启动，从子程序返回。

m=1: 从子程序返回, 并存储条件启动。

n=0: 不存储中断, 从子程序#COMINT 返回。

n=1: 从通信中断#COMINT 子程序返回, 并存储中断。

注 1: 此参数的缺省值为 0。例如，EN，1 和 EN0，1 具有相同的作用。

注 2: 此参数规定了#COMINT 子程序处理条件启动的方法。条件启动使得程序等待特殊事件产生。例如, AM 命令等待所有轴运动完成。如果程序正处于等待条件启动中, 由于通信中断而执行#COMINT 子程序, #COMINT 就会结束, 转而继续等待条件启动或清除条件启动。在条件启动之后的命令处继续执行程序。

注 3: 使用 RE 命令从中断处理子程序#LIMSWI 和#POSERRR 返回。使用 RI 命令以 #ININT 子程序返回。

用途:		缺省:
运动进行中	Yes	缺省值 m=0, n=0
在程序中	Yes	缺省格式
命令行	No	
适用控制器	所有型号控制器	

相关命令:

“RE”	从报警子程序返回
“RI”	从中断子程序返回

举例:

#A	程序 A
PR 500	A 轴正向运动 500 计数单位
BGA	开始 A 轴运动
AMA	暂停程序直到 A 轴运动完成
PR 1000	A 轴正向运动 1000 计数单位
BGA	开始运动
EN	程序结束

注意: 取代 EN, 用 RE 命令来结束报警子程序和限位子程序。使用 RI 命令来结束输入中断子程序。

59) ENDIF

功能: IF 条件语句结束

说明: 用 ENDIF 命令来表明 IF 条件语句的结束。IF 条件语句由 IF 和 ENDIF 命令来构成。对于已执行的每条 IF 命令都必须执行一条 ENDIF 命令。建议用户不要在 IF 条件语句里含有跳转命令, 否则会引起命令执行的死循环。在这种情况下, 就不可能执行 ENDIF 命令。

参数: ENDIF

用途:

运动进行中	Yes
在程序中	Yes
命令行	No
适用控制器	所有型号控制器

相关命令:

“II (二进制 EC)”	输入中断
“ELSE”	只在 IF 命令之后使用的选择命令
“JP”	跳转命令
“JS”	跳转到子程序的命令

举例:

IF (@IN[1]=0)	以 INPUT 1 为基础的 IF 条件语句
MG “INPUT 1 IS ACTIVE”	如果 “IF” 条件为假, 发送信息
ENDIF	条件语句结束

60) EO

功能： 回声

说明： EO 命令使回声开或关。如果回声为关，由总线进行字符输入就不会有回声。

参数： EO n

其中： n=0 回声关
 n=1 回声开

用途：

运动进行中	Yes	缺省：	缺省值	0
在程序中	Yes		缺省格式	1.0
命令行	Yes			
适用控制器	除 Ethenet 通信外的所有型号控制器			

举例：

EO 0 关回声
EO 1 开回声

61) EP (二进制 C2)

功能： 凸轮表间隔及起始点

说明： EP 命令定义 ECAM 表间隔和偏置值。偏置是第一个 ECAM 表入口的主动轴位置。间隔是 2 个相关表入口之间的主动轴位置的差值。此命令有效地定义了 ECAM 表的尺寸大小。参数 m 是间隔，而 n 是起始点，可以指定 257 个点。

参数： EP m, n

其中： m 为正整数，范围为 1~32767。
m=? 返回间隔 m 值。
n 是-214783648~2147483647 之间的整数，n 是偏置值。

用途：

运动进行中	Yes	缺省：	缺省值
在程序中	Yes		缺省格式
命令行	Yes		
适用控制器	所有型号控制器		

操作应用：

_EP 读取间隔 m 值

相关命令：

“EA”	选择 ECAM 主动轴
“EB (二进制 C4)”	使 ECAM 有效
“EC (二进制 C6)”	设置 ECAM 表标志
“EG (二进制 C3)”	启动 ECAM
“EM (二进制 C1)”	指定 ECAM 周期
“EQ (二进制 C5)”	ECAM 停止
“ET (二进制 C0)”	ECAM 表

举例：

EP 20, 100 设置凸轮主动轴点为 100, 120, 140……
D=_EP 设置变量 D 等于 ECAM 间隔值

62) EQ (二进制 C5)

功能： ECAM 退出

说明：EQ 命令使电子凸轮从动轴在所指定的主动轴位置退出。能够对每个轴指定分离点。如果将此值指定在主动轴范围之外，从动轴就立即退出。

参数：EQ n, n, n, n, n, n, n, n 或 EQA=n

其中：n 是要退出轴的主动轴位置。

n=? 如果发布啮合命令且该轴正等待啮合，返回值为 1，如果发布退出命令且该轴正等待退出，返回值为 2，如果 ECAM 已啮合或已退出，返回值为 0。

用途：

运动进行中 Yes
在程序中 Yes
命令行 Yes

适用控制器 所有型号控制器

缺省：

缺省值
缺省格式

操作应用：

_EQn 如果发布啮合命令且该轴正等待啮合，返回值为 1，如果发布退出命令且该轴正等待退出，返回值为 2，如果 ECAM 已啮合或已退出，返回值为 0。

相关命令：

“EA”	选择 ECAM 主动轴
“EB (二进制 C4)”	使 ECAM 有效
“EC (二进制 C6)”	设置 ECAM 表标志
“EG (二进制 C3)”	启动 ECAM
“EM (二进制 C1)”	指定 ECAM 周期
“EP (二进制 C2)”	指定 ECAM 表间隔和起始点
“ET (二进制 C0)”	ECAM 表

举例：

EQ 300, 700 A 轴和 B 轴电机分别在主动轴位置 300 和 700 处退出。

注意：此命令不是条件启动命令。此命令不会使程序流持续执行。若需要使执行连续到达主动轴位置，就使用 MF 或 MR 命令。

63) ER (二进制 88)

功能：误差限制

说明：ER 命令为各轴设定允许的位置误差幅值。当超过此极限值时，误差报警输出就变低(超差)。如果超差关断 (OE 1) 命令有效，电机使能就会关断。

参数：ER n, n, n, n, n, n, n, n 或 ERA=n

其中：n 是 1~32767 之间的无符号数，代表误差极限值，以编码器计数单位表示。

-1 值会使所指定轴的位置误差极限无效。

n=? 返回指定轴的误差极限值。

用途：

运动进行中 Yes
在程序中 Yes
命令行 Yes

适用控制器 所有型号控制器

缺省：

缺省值 16384
缺省格式 位置格式

操作应用：

_ERn 读取所指定轴的位置误差极限值。

相关命令：

“OE (二进制 8D)” 超差关断

#POSERR

超差自动处理子程序

举例:

ER 200, 300, 400, 600

设定 A 轴误差限制值为 200, B 轴为 300,
C 轴为 400, D 轴为 600

ER , 1000

设定 B 轴误差极限值为 1000, 保留 A 轴原值不变

ER ? , ? , ? , ?

返回 A, B, C, D 轴误差值

00200, 01000, 00400, 00600

ER ?

返回 A 轴误差值

00200

V1= _ERA

为 ERA 的值分配变量 V1

V1=

返回 V1 值

00200

提示: 用 ER 指定的误差极限应该足够大, 以便在正常运行期间也不会超出。超出误差极限时, 会使机械卡住或损坏系统文件, 如编码器或放大器等。

64) ES

功能: 椭圆缩放

说明: 在矢量方式 (XM) 中, ES 命令将某一轴的分辨率进行分割。当编码器分辨率不同时, 用此命令仍可产生圆弧运动。也可产生椭圆而不是圆弧。

此命令有两个参数 m 和 n, 用于由命令 VM 所指定的轴。当 $m > n$ 时, 第一轴 (X) 的分辨率将由比例 m/n 来分割。分辨率变更选用于 VP 和 CR 命令, 它会对较高分辨率的轴进行有效改变, 以便与 coarser 分辨率相匹配。

ES 命令适用于所选择的坐标系, S 或 T。要选择坐标系, 请使用命令 CAS 或 CAT。

参数: ES m, n

其中: m 和 n 是 1~65535 之间的正整数

用途:

运动进行中 Yes

在程序中 Yes

命令行 Yes

适用控制器 所有型号控制器

缺省:

缺省值 1, 1

缺省格式

相关命令:

“VM”

矢量方式

“CR (二进制 B3)”

圆弧运动

“VP (二进制 B2)”

矢量位置

举例:

VMAB; ES 3, 4

将 B 轴分辨率用 4/3 来分割

VMCA; ES 2, 3

将 A 轴分辨率用 3/2 来分割

VMAC; ES 3, 2

将 A 轴分辨率用 3/2 来分割

65) ET (二进制 CO)

功能: 电子凸轮表

说明: ET 命令为从动轴设置 ECAM 表入口, 不需要主动轴的值, 当主动轴处于点 $(n \times i) + o$ 处时, 从动轴入口 (n) 是从动轴的位置, 其中 i 是间隔, o 是由 EP 命令所决定的偏置。

参数： ET[n]=n, n, n, n, n, n, n, n, n

其中：n 是-2147438648~2147438647 之间的整数，如果标志计数已经用命令 EC 加以设定，n 值就可放在命令之外，在此方式中，每个 ET 命令就将标志计数自动加 1。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值
缺省格式

相关命令：

“EA”	选择 ECAM 主动轴
“EB（二进制 C4）”	使 ECAM 有效
“EC（二进制 C6）”	设置 ECAM 表标志
“EG（二进制 C3）”	启动 ECAM
“EM（二进制 C1）”	指定 ECAM 周期
“EP（二进制 C2）”	指定 ECAM 表间隔和起始点
“EQ（二进制 C5）”	ECAM 停止

举例：

ET[0]=0,, 0	指定从动轴 A 和 C 与主动轴起始点相同步的位置
ET[1]=1200,, 400	指定从动轴 A 和 C 与主动轴第二点相同步的位置
EC0	设定凸轮表标志值为 0，表中第一个元素。
ET0,, 0	指定从动轴 A 和 C 与主动轴起始点相同步的位置
ET 1200,, 400	指定从动轴 A 和 C 与主动轴起始点相同步的位置

66) FA（二进制 94）

功能： 加速度前馈

说明： FA 命令设定加速度前馈系数。当加速度相乘时，在加速期间此系数增加转矩偏置电压，在减速期间，减小转矩偏置电压。

加速度前馈偏置=FA×AC×1.5×10⁻⁷

减速度前馈偏置=FA×DC×1.5×10⁻⁷

前馈偏置乘积限制在 10V 以内，当以 PA, PR 和 JG 命令运动时，FA 工作。

参数： FA n, n, n, n, n, n, n, n, n 或 FAS=n

其中：n 是 0~8192 之间的无符号十进制数，分辨率为 0.25。

n=? 返回所指定轴前馈加速度的数值。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值 0
缺省格式 4.0

操作应用：

_FAn 读取所指定轴前馈加速度的数值

相关命令：

“FV（二进制 95）” 速度前馈

举例：

AC 500000, 1000000

FA 10, 15	设定 A 轴前馈系数为 10, B 轴为 15,
	即 A 轴有效偏置为 0.75V, B 轴为 2.25V
FA ? , ?	查询 A, B 轴前馈系数
010, 015	

注意: 如果在运动当中更改前馈系数, 到下一次运动时, 此改变才会起作用。

67) FE (二进制 A4)

功能: 寻找档块前沿

说明: FE 命令使电机运动直到检测到该轴原点输入开关状态改变。运动的方向取决于原点输入的初始化状态 (用 CN 命令来配置原点输入的极性)。一旦检测到原点输入开关状态变化, 电机就减速停止。

此命令对于创建用户自己的原点程序很有用。

参数: FE nnnnnnnn

其中: n 是 A, B, C, D, E, F, G, H 或指定轴的任意组合, 无参数则指定所有轴。

用途:

运动进行中	No
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值
缺省格式

相关命令:

“FI (二进制 A5)”	寻找标志脉冲
“HM (二进制 A3)”	原点返回
“BG (二进制 A0)”	开始运动
“AC (二进制 90)”	加速度
“DC (二进制 91)”	减速度
“SP (二进制 92)”	速度

举例:

FE	设定寻找档块方式
BG	开始各轴运动
FEA	只寻找 A 轴档块
BGA	
FEB	只寻找 B 轴档块
BGB	
FECD	寻找 C、D 轴档块
BGCD	

提示: 寻找档块只搜寻原点开关输入点状态的变化。要搜寻编码器标志脉冲, 请使用 FI 命令。要想搜寻原点开关输入和标志脉冲, 请使用 HM 命令。请记住: 在每条这样的命令之后, 要指定 BG 命令。

68) FI (二进制 A5)

功能: 寻找标志脉冲

说明: FI 命令和 BG 命令使电机运动, 直至检测到编码器标志脉冲。控制器捕捉标志脉冲从低到高变化。当检测这一变化时, 运动停止, 此位置被视为零位。为了提高回零精度, 搜寻标志期间的速度应该指定在 500 计数单位/sec 之下。FI 命令对于编写专用原点返

回程序很有用。运动方向由 JG 命令的符号来指定。

参数: FI nnnnnnnn

其中: n 是 A, B, C, D, E, F, G, H 或所指定轴的任意组合, 无参数意味着指定所有轴。

用途:

运动进行中	No
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值
缺省格式

相关命令:

“FE (二进制 A4)”	寻找档块前沿
“HM (二进制 A3)”	原点返回
“BG (二进制 A0)”	开始运动
“AC (二进制 90)”	加速度
“DC (二进制 91)”	减速度
“SP (二进制 92)”	速度

举例:

#HOME	原点返回程序
JG 50	设定 JOG 速度, 正向
FIA	寻找标志脉冲
BG	开始运动
AMA	运动完成后
MG “FOUND INDEX”	

提示: FI 只搜寻标志脉冲的状态变化。要想搜寻原点开关状态变化, 请使用 FE 命令。要搜寻原点开关输入和标志脉冲状态变化, 请使用 HM 命令。请记住在每条命令之后要指定 BG。

69) FL (二进制 8E)

功能: 正向软限位

说明: FL 命令设置正向软件位置极限值。如果在运动期间超过此极限, 该轴运动就会减速停止。超过此极限的正向运动被禁止。正向限位在 A+1, B+1, C+1, D+1 处生效。正向限位在 2147483647 时无效。

当负向软件限位生效时, 如果自动子程序#LIMSWI 写入程序中就会执行#LIMSWI。参见《用户手册》中自动子程序说明。

参数: FL n, n, n, n, n, n, n 或 FLA=n

其中: n 是-2147483648~2147483647 之间的带符号整数。n 代表运动轴的绝对位置。

n=2147483647 使正向限位关断。

n=? 返回所指定轴的正向限位开关值。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	2147483647
缺省格式	位置格式

操作应用:

_FLn 读取所指定轴的正向软限位值。

相关命令：

“BL (二进制 8F)”	负向软限位
“PF”	位置格式

举例：

FL 150000	设定 A 轴正向软限位为 150000 计数单位
#TEST	程序名
AC 1000000	加速度
DC 1000000	减速度
FL15000	正向软限位值
JG 5000	JOG 正向运动
BGA	开始运动
AMA	限位后停止
TPA	响应位置
EN	结束

提示： GALIL 控制器也提供硬件限位。

70) FV (二进制 95)

功能： 速度前馈

说明： FV 命令设置速度前馈系数，或返回先前的设定值。此系数产生与命令速度成正比的输出偏置信号。

速度前馈偏置 = $1.22 \times 10^{-6} \times \text{FV} \times \text{速度} [\text{cts/s}]$

当用 PA, PR, JG, VM, LM, CM 命令运动时，FV 会起作用。

例如：如果 FV=10，速度为 200000cts/sec，速度前馈偏置就等于 2.44V。

参数： FV nnnnnnnn 或 FVA=n

其中：n 是 0~8191 之间的十进制无符号数。

n=? 返回指定轴的前馈速度。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	3.0

操作应用：

_FVn 读取指定轴的前馈速度

相关命令：

“FA (二进制 94)”	加速度前馈
---------------	-------

举例：

FV 10, 20	设定 A 轴、B 轴前馈系数分别为 10 和 20
JG 30000, 80000	A 轴为 0.366V, B 轴为 1.95V
FV ? , ?	查询 A, B 轴 FV 值
010, 020	

71) GA

功能： 电子齿轮主动轴

说明：GA 命令为电子齿轮指定主动轴。可以指定多个主动轴。主动轴可以是主编码器输入、辅编码器输入或任意轴的命令位置。主动轴也可以是以 LM 或 VM 坐标转动的矢量运动轴运动。当主动轴是一个命令的矢量运动时，将矢量运动视为正向，如果电子齿轮比为正，则从动轴将以正向运动；如果电子齿轮比为负，则从动轴以反向运动。由 GR 命令指定从动轴和齿轮比，用命令 GR 使电子齿轮关断。

参数：GA n, n, n, n, n, n, n, n 或 GAA=n

其中：n 可以是 A, B, C, D, E, F, G, H 或 N。X 值被用来将指定的主编码器轴设置为电子齿轮主动轴，N 代表虚拟轴。从动轴由参数的位置加以指定。参数的第一个位置对应于‘A’轴，第二个位置对应于‘B’轴等等。如果所对应的轴不作为从动轴，就必须在参数处使用逗号。

n 可以是 CA, CB, CC, CD, CE, CF, CG 或 CH。X 值被用来将指定轴的命令位置设置为电子齿轮主动轴。

n 可以是 S 或 T。用 S 或 T 将 S 坐标系或 T 坐标系中的矢量运动规定为电子齿轮主动轴。

n 可以是 DA, DB, DC, DD, DE, DF, DG 或 DF。用 n 值将指定的辅编码器轴设置为电子齿轮主动轴。

用途：

运动进行中	No
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值
缺省格式

相关命令：

“GR (二进制 96)”	齿轮比
“GM”	龙门同步方式

举例：

#GEAR	电子齿轮程序
GA , A, T	指定 A 轴作为 B 轴的主动轴， T 坐标矢量运动作为 C 轴的主动轴
GR , 0.5, -2.5	指定 B 轴和 C 轴齿轮比
JG 5000	指定主动轴 JOG 速度
BGA	开始运动
WT 10000	等待 10000msec
STA	停止

提示：用命令位置作为主动轴对龙门同步应用非常有用。用矢量运动作为主动轴在产生螺旋线运动方面非常有用。

72) GM

功能：龙门同步方式

说明：GM 命令指定执行电子齿轮的轴以龙门同步方式运动，在此方式中，不用 ST 命令或限位开关来停止齿轮运动，只有 GR0 会使此方式中的齿轮停止。

参数：GM n, n, n, n, n, n, n, n 或 GMA=n

其中：n=0 没有龙门同步功能

n=1 使龙门同步方式使能

n=? 返回所指定轴的龙门同步方式的状态：0=龙门同步方式无效，1=龙门同步方式

使能

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	0
缺省格式	1.0

操作应用:

_GMn 读取指定轴龙门同步方式的状态: 0=龙门同步方式无效, 1=龙门同步方式使能。

相关命令:

“GR (二进制 96)”	齿轮比
“GA”	电子齿轮主动轴

举例:

GM 1, 1, 1, 1	对所有轴 GM 使能
GM0	A 轴 GM 无效, 其它轴保持不变
GM , , 1, 1	对 C, D 轴 GM 使能, 其它轴保持不变
GM 1, 0, 1, 0	对 A, C 轴 GM 使能, B, D 轴 GM 无效

提示: GM 功能对驱动双边重负载 (龙门型负载) 非常有用。

73) GR (二进制 96)

功能: 电子齿轮比

说明: GR 命令为电子齿轮方式中的齿轮轴指定齿轮比。用 GA 命令定义主动轴, 各齿轮轴的齿轮比可以不一样。主动轴能够以两个方向旋转, 齿轮比为零则使各轴的电子齿轮无效。限位开关也使电子齿轮无效, 但龙门同步方式使能时除外 (见 GM 命令)。

参数: GR n, n, n, n, n, n, n, n 或 GRA=n

其中: n 是 ±127 之间的带符号数, 分辨率为 0.0001。

n=0 使电子齿轮无效。

n=? 返回指定轴的齿轮比值。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	0
缺省格式	3.4

操作应用:

_GRn 读取指定轴的电子齿轮比值

相关命令:

“GA”	主动轴
“GM”	龙门同步方式

举例:

#GEAR	
MOB	关断 B 轴伺服使能
GAB	指定 B 轴为主动轴
GR 0.25, , -5	指定 A 轴和 C 轴齿轮比
EN	程序结束

此时，当用手转动 B 轴电机时，A 轴会以 1/4 的速度旋转，而 C 轴会以反方向 5 倍速度旋转。

提示：当电子齿轮轴必须与主动轴刚性耦合时，请使用龙门同步方式（GM）。

74) HM (二进制 A3)

功能：原点返回

说明：对于伺服系统而言，HM 命令执行三级原点返回动作，而对于步进电机来说，执行二级原点返回动作。

对于伺服电机工作：在原点返回程序的第一阶段，电机以用户编程速度运动直至检测到该轴原点开关输入状态变化。第一阶段的运动方向由原点开关输入的初始化状态来决定。一旦原点开关输入改变状态，电机就减速停止。原点开关输入的状态能够用 CN 命令进行配置。

在第二阶段，电机改变方向，并再次慢速副近原点开关，当检测原点开关时，电机立即停止。

在第三阶段，电机慢速向前运动，直至检测到来自于编码器的标志脉冲。电机停在此点，并将此点位置定义为零位。

对于步进电机原点返回，只有第一、二阶段，第二阶段运动的指令脉冲频率为 256cts/sec.

用途：

运动进行中	No
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值
缺省格式

操作应用：

_HMn 读取所指定轴的原点开关状态

相关命令：

“CN (二进制 E6)”	配置原点返回
“FI (二进制 A5)”	只搜寻标志脉冲
“FE (二进制 A4)”	只搜寻原点开关

举例：

HM	对所有轴设定原点返回方式
BG	所有轴原点返回
BGA	只有 A 轴原点返回
BGB	只有 B 轴原点返回
BGC	只有 C 轴原点返回
BGD	只有 D 轴原点返回

提示：您可以用 FE 和 FI 命令来创建您自己要用的原点返回程序。

75) HS

功能：端口分配开关

说明：用 HS 命令来转换两个端口之间的端口分配。当端口用 HC 命令打开时，由控制器分配端口，或用 IH 命令明确进行分配。若需要修改这些分配，HS 命令就使端口重新分配。

参数：HSh=I

其中：h 是第一端口转换 (A~H, S)

I 是第二端口转换 (A~H, S)

S 用来代表正在执行命令的当前端口。

用途:		缺省:	
运动进行中	Yes	缺省值	-----
在程序中	Yes	缺省格式	-----
命令行	Yes		
适用控制器	DMC-2100, DMC-2200		

相关命令:	
“IH”	以太网端口
“HC”	自动端口连接
“HR”	端口恢复

举例:	
HSC=D	将端口 C 的连接分配到端口 D。端口 D 的连接分配给端口 C。
HSS=E	将正在执行的端口分配给端口 E。端口 E 的连接分配给正在执行的端口。

76) HX

功能: 暂停执行

说明: HX 将运行中的任何程序暂停执行。

参数: HXn

其中: n 是 0~7 之间的整数, 它表示域号。

用途:		缺省:	
运动进行中	Yes	缺省值	n=0
在程序中	Yes	缺省格式	
命令行	Yes		
适用控制器	所有型号控制器		

操作应用:	
当用作运算时, _HXn 会有域 n 的运行状态:	
0	未运行的域
1	域正在运行
2	域已停止在条件启动点

相关命令:	
“XQ”	执行程序
“ST (二进制 A1)”	停止运动的所有域

举例:	
XQ#A	执行程序#A, 0 域
XQ#B, 3	执行程序#B, 3 域
HX0	暂停 0 域
HX3	暂停 3 域

77) IA

功能: IP 地址

说明: IA 命令给控制器分配 IP 地址;

只有使用 TCP/IP 协议时, 此命令才可以指出时间溢出值;

IA 命令只能通过 RS-232 串口才能使用, 因为它分配 IP 地址给控制器, 且与控制器

的通讯只有设好 IP 地址后才能通过 Internet 实现。

参数: IA ip0, ip1, ip2, ip3 或 IA n 或 IA < t

其中: ip0, ip1, ip2, ip3 是 1 字节数, 通过逗号隔开, 表示 IP 地址各自的领域;

n 表示控制器的 IP 地址是一个 32 位带符号整数 (2 的补码);

< t 表示重设 TCP 时的最新采样时间 (只有 TCP/IP 连接时);

> u 表示多点传送 IP 地址, 其中 u 是 0~63 之间的整数 (只有 UDP/IP 连接时);

IA? 返回控制器的 IP 地址

用途:

运动进行中

No

在程序中

No

命令行

Yes

适用控制器

DMC-2100, DMC-2200

缺省:

缺省值

n=0, t=250

缺省格式

操作应用:

_IA0 读取 IP 地址, 表示 32 位符号数 (2 的补码)

_IA1 读取 t 值 (重试时间)

_IA2 读取有效端口数

_IA3 读取可操作端口数, 而这个数为 0~5 之间的数。0 表示端口 A, 1 表示端口 B 等等。

_IA4 读取失去通讯时的最后端口数, 复位时读取 A-1, 指出无端口丢失

_IA5 读取返回以太网的速度, 10 为 10Base T, 100 为 100Base T*

*只用于 DMC-2200 时才有效。

相关命令:

IH

Internet 端口

举例:

IA 151, 12, 53, 89

控制器的 IP 地址为 151, 12, 53, 89

IA 2534159705

控制器的 IP 地址为 151, 12, 53, 89

IA <500

设置时间溢出值为 500msec

78) IF

功能: IF 条件语句 (假设)

说明: 在 IF 条件语句中, IF 命令必须与 ENDIF 命令组成一个完整的程序格式。而这个条件可以为 1 个或者更多个, 但每个条件结束后必须用小括号隔开。如果条件是真实的, 那么命令解释程序将随着 IF 命令继续执行下去, 反之若条件是假的, 控制器将忽略命令, 直到在程序中出现相关的 ENDIF、OR、ELSE 命令为止。

参数: IF (条件)

其中: 测试条件时的逻辑符号如下所示:

< 小于

> 大于

= 等于

<= 小于或等于

> = 大于或等于 < > 不等于

注：位方式操作 | 及&可用于测试多个条件。

用途：

运动进行中	Yes
在程序中	Yes
命令行	No
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	-

相关命令：

“ELSE”	用于 IF 命令后，表示选择
“ENDIF”	结束 IF 条件

举例：

IF (_TEA <1000)	如果条件为 A 电机位置
MG “Motor is within 1000 counts of zero ”	若 “IF” 条件为真，发送信息
ENDIF	IF 条件语句结束

79) IH

功能： 开放式互联网端口

说明： 当 DMC-2100 作为主控制器工作时，可以用 IH 命令。此命令可打开 DMC-2100 端口且连接到从控制器上。

在任何时候，每个控制器可以有 6 个端口开放，分别用英文字母 A, B, C, D, E, F 表示。要想打开端口，用户必须指出以下几点：

- A. 从控制器的 IP 地址
- B. TCP/IP 或 UDP/IP
- C. 从控制器的通信口数。如果从设备不要求具体的通信口值时，可以不用这个数。若未指出这个数，此时通信口值默认为 1000。

参数： IHh= ip0, ip1, ip2, ip3 <P> q 或 IHh=n <P> q 或 IHh=> r

其中：h 是端口，用 A, B, C, D, E, F 表示；

ip0, ip1, ip2, ip3 是 0~255 之间的整数，表示 IP 地址各自的领域，每个值之间需用逗号隔开；

n 是-2147483648~2147483648 之间的带符号整数。此值是 32 位 IP 地址，可以指定 4 个地址领域来替换使用；

IHS=> r 发送命令时关闭端口。其中 r=-1 时为 UDP/IP，r=-2 时为 TCP/IP；

IHT=> r 除发送命令的端口外，其余的端口都关闭。若 where r=-1 时为 UDP，若 r=-2 时为 TCP；

<p 指出从控制器的通信口数，其中 p 是 0~65535 之间的整数。此值不要求打开端口；

> q 指出连接类型，其中 q 是 0 时表示没有连接，是 1 时为 UDP 型，2 为 TCP 型；

> r 表示终止连接且释放端口，其中 r=-1 时为 UDP，r=-2 时为 TCP；

“?” 返回 IP 地址，用 4 个 1 字节数表示

操作应用：

_IHh0 读取 32 位数的 IP 地址

_IHh1 读取从通信口数

_IHh2 若释放端口是空的，则为 0

若是从 UDP，则为 1

若是从 TCP，则为 2

若是主 UDP，则为-1

若是主 TCP，则为-2

若想建立 UDP 端口时，则为-5

若想建立 TCP/IP 端口时，则为-6

_IHh3 若 ARP 成功，则为 0

若失败或运行中，则为 1

用途：

运动进行中

No

在程序中

Yes

命令行

Yes

适用控制器

DMC-2100, DMC-2200

缺省：

缺省值

缺省格式

相关命令：

“IA”

Internet 地址

举例：

IHA=251, 29, 51, 1

当 IP 地址为 251.29.51.1 时打开端口 A

IHA=-2095238399

当 IP 地址为 251.29.51.1 时打开端口 A

注：当给出 IH 命令时，控制器在打开端口前初始化从控制器上的 ARP。这个操作在控制器响应前会导致一小段时间的延误。

80) II (二进制 EC)

功能：输入中断

说明：II 命令能够对输入进行中断。当缺省时，逻辑“0”是有效的，但也能设置成逻辑“1”。如果在执行程序时，任何指定的输入都有效，那么程序将跳到#ININT 子程序中，而设置的一些条件启动也将被清除，但可以用 RI 命令来终止中断子程序，并从#ININT 子程序返回。

参数：II m, n, o, p

其中：m 是 0~8 之间的十进制整数。若为 0，中断无效。m 值指定了用于输入中断时的最小输入值。当删除第二个参数 n 时，只有 m 指定的输入使能有效。

n 是 2~8 之间的整数。这个参数具有可选性，它可以与 m 参数共同指定一定范围内的输入中断。例如，II 2, 4 表明中断发生在输入 2，输入 3，输入 4。

o 是 1~255 之间的整数。它可以与 m、n 交替使用去指出输入中断的范围。若指出了 m、n 参数，o 参数将被忽略。o 是一个整数值，用二进制数表示。例如，若 o=15 时，用二进制表示则是 00001111，它的后 4 位是 1（从位 0 到位 3）而前 4 位则是 0（从位 4 到位 7）。每位代表 1 个使能中断，如位 0 表示中断 1，位 1 表示中断 2 等。若 o=15，输入 1, 2, 3, 4 都有效。

P 是 1~255 之间的整数。当输入为逻辑“1”时，p 参数有效。这个参数是个整数值，

用二进制表示。这个二进制数使用“AND”去连接 m、n 或 o 参数所指定的输入。例如，如果 m=1，n=4 时，输入 1，2，3，4 都有效。若 p=2 时（用二进制表示为 00000010），输入 2 在逻辑‘1’时有效，反之，在逻辑“0”时输入 1，3，4 有效。

用途：

运动进行中	Yes
在程序中	Yes
命令行	No
适用控制器	所有型号控制器

缺省：

缺省值	
缺省格式	3.0（只有屏蔽）

相关命令：

“RI”	从中断点返回主程序
#ININT	中断子程序
“AI”	在输入信号到达之后启动

举例：

#A	程序 A
II1	指定在输入 1 中断
JG5000; BGA	指定 JOG 方式且启动 A 轴
#LOOP; JP#LOOP	循环
EN	结束程序
#ININT	中断子程序
STA; MG “INTERRUPT”; AMA	停止 A 轴，打印信息，等待运动完成
#CLEAR; JP#CLEAR, @IN[1]=0	检查中断清除
BGA	开始运动
RI0	返回到主程序，条件启动不能重新使能

81) IL (二进制 89)

功能：积分极限

说明：IL 命令指出滤波器的积分极限在一定的电压范围内有效。例如，IL2 说明了 A 轴的积分输出极限为+/-2V 之间。

若参数为负数时，积分输出在运动时的电压保持不变。例如，IL-3 说明了积分输出极限为+/-3V 之间。但若运动开始时，积分输出为 1.6V，那么在整个运动过程中，积分输出始终为 1.6V。注意，在任何情况下，KD、KP 命令都有效。

参数：Il n, n, n, n, n, n, n, n 或 ILA= n

其中：n 是+/-10V 之间的数，分辨率为 0.0003。

n=? 返回指定轴的积分极限值

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	9.9988
缺省格式	1.4

操作应用：

_ILn 读取指定轴的积分极限值

相关命令：

“KI”	积分常数
------	------

举例：

KI 2, 3, 5, 8	积分常数
IL 3, 2, 7, 2	积分极限
IL?	返回 A 轴极限
3.0000	

82) IN**功能：**输入变量

说明：使用 IN 命令时，用户可以通过键盘输入变量。当程序执行 IN 命令时，会显示提示信息。此时用户敲回车后就可以输入新的变量值，然后再给新的变量值定义一个变量名。使用 IN 命令时，将会使后续命令的执行挂起，直到检测到回车或引号。若没有输入新变量值，那么程序将按老变量值运行。输入中断、错误中断及限位开关中断都有效。IN 命令只有在域 0 时使用才有效。

注：当使用 DMC2000 或 DMC2100 时，控制器只有带通信串口使用 IN 命令才有效。

参数：IN “m”， n

其中：m 是提示信息

n 是变量名

m 和 n 的字符之和必须少于 80 个字符。

用途：

运动进行中	Yes
在程序中	Yes
命令行	No
适用控制器	所有型号控制器

缺省：

缺省值	-----
缺省格式	位置格式

举例：（用户定义长度单位是：英寸，速度单位是：英寸/秒（丝杆螺距为 2，编码器计数为 2000/转）

#A	程序 A
IN “Enter Speed (in/sec)”, V1	提示速度操作 V1
IN “Enter Length (in)”, V2	提示长度操作 V2
V3=V1*4000	转换单位为计数/秒
V4=V2*4000	转换单位为计数
SP V3	速度
PR V4	位置
BGA	运动开始
AMA	等待运动完成
MG “MOVE DONE”	打印信息
EN	程序结束

83) IP**功能：**增量位置

说明：当电机运动时，IP 命令可以改变命令位置。此命令不要求 BG。根据所执行的运动，此命令会产生三种情况，这是个求四倍频数。

第一种情况：电机静止时

IP a, b, c, d 与 PR a, b, c, d 及 BG 命令相等。电机将移动到转速及加速度所要求的位置。

第二种情况：电机朝着 PR, PA, 或 IP 指定的位置移动。

IP 命令将会使电机朝着新目标位置移动，而这个新目标位置是老位置加增量位置。增量位置必须与当前的运动保持相同方向。

第三种情况：电机处于 Jog 模式下

IP 命令将会使电机移动到当前瞬时位置加指定的增量式位置。SP 参数及 AC 参数无效。当两轴同步，其中 1 轴速度由于轮距的变化而不确定时，使用此命令非常有用。

警告：当为 Jog 模式时，IP 会产生瞬时位置误差。因此，IP 命令只适用于小增量位置移动。

参数： IP n, n, n, n, n, n, n, n, n 或 IPA= n

其中： n 是-2147483648~2147483647 之间的十进制带符号数。

n=? 返回指定轴的当前位置

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	
缺省格式	7.0

相关命令：

“PF” 位置格式

举例：

IP50	设置加速度及速度为 50 计数
#CORRECT	标号
AC100000	设置加速度
JG10000; BGA	在 Jog 模式下，以 10000 计数/秒的速率启动 A 轴
WT1000	等待 1000 msec
IP10	电机立刻移动 10 计数
STA	停止运动

84) IT (二进制 93)

功能：独立时间常数——平滑操作

说明：IT 命令对独立运动中的加、减速度函数进行滤波以产生平滑速度包路线。最终的包路线，也称为平滑，可以连续加速度并使机械振动减小。IT 设置滤波器的带宽，其中 1 表明无滤波，0.004 意味着最大滤波。注意：平滑滤波会导致运动时间延长。

使用 IT 命令，不会影响条件启动 AR、AD。因为条件启动 AR、AD 在 IT 滤波前控制包路线，所以如果 IT 不为 1 时，能满足在实际距离前到达。

参数： I1 n, n, n, n, n, n, n, n, n 或 ITA= n

其中： n 是 0.004~1.0 之间的正数，分辨率为 1/256。

n=? 返回指定轴独立时间常数值。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	1
缺省格式	7.0

操作应用：

—ITn 读取 ‘n’ 轴独立时间常数值。

相关命令：

“VT”

矢量时间平滑常数

举例：

IT0.8, 0.6, 0.9, 0.1
IT?
0.8
1.

a, b, c, d 轴设置独立时间常数
返回 A 轴独立时间常数

85) JG (二进制 A8)

功能：Jog

说明：设置 jog 模式及 jog 转速。

参数：Il n, n, n, n, n, n, n, n 或 JGA= n

其中： n 是 0~+/-12, 000, 000 之间的十进制带符号数，此值单位是计数/秒。

当为步进电机操作时，最大值为 3, 000, 000 步/秒。

n=? 返回指定轴的 jog 速度绝对值。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	16385
缺省格式	位置格式

操作应用：

—JGn 读取指定轴的 jog 速度绝对值。

相关命令：

“BG”	开始
“ST”	停止
“AC”	加速度
“DC”	减速度
“IP”	增量位置
“TV”	响应速度

举例：

JG 100, 500, 2000, 5000

在 Jog 模式下，设置 A 轴转速为 100 计数/秒，B 轴为 500 计数/秒，C 轴为 2000 计数/秒，D 轴为 5000 计数/秒。

BG

开始运动

JG, , -2000

改变 C 轴方向，让它运行到-2000 计数/秒

86) JP

功能：跳到指定的程序

说明：在规定条件下，JP 命令导致 1 个跳动到程序入口。程序入口可以是行号或标号 (#)，而条件必须是使用一个逻辑操作例如等于或小于的条件语句。如果指定的条件满足，程序方能跳动。

在单个跳动程序中可以使用多个条件，且这些条件语句可以用“&”或“|”来连接。其中，“&”用于连接任意两个条件，但两个条件必须都是真实的；而“|”只要求两个条件中的一个条件是真实的就可以。注：控制器会检测圆括号中的每个条件。

参数：JP 入口，条件

其中：入口是程序的行号或标号；而条件是使用逻辑操作的条件语句。

具体的逻辑操作符号如下：

〈 小于
〉 大于
= 等于
〈= 小于或等于
〉= 大于或等于
〈〉 不等于

用途：

运动进行中	Yes
在程序中	Yes
命令行	No
适用控制器	所有型号控制器

缺省：

缺省值
缺省格式

相关命令：

“JS”	跳到子程序
“IF”	条件
“ELSE”	逻辑判断语句
“ENDIF”	结束逻辑判断

举例：

JP#POS1, V1 〈5	如果变量 V1 小于 5 时，跳到 POS1
JP#A, V7*V8=0	如果 V7 乘以 V8 等于 0 时，跳到#A
JP#B	跳到#B（无条件）

提示：JP 跟 IF、THEN 命令类似。当程序运行逗号右边的条件时，则会发生跳动，而逗号前面的标号位置则是程序跳到的位置。

87) JS

功能：跳到子程序

说明：JS 命令将改变程序中所执行命令的顺序。如果执行了跳动，程序将一直运行下去，除非遇到行号或标号的程序语句时才会停下来。程序在 JS 命令的提示下，继续执行，在遇到下一个 EN 命令（结束子程序）后，保存行号。在子程序中必须含有 JS 命令。

在单个跳动程序中可以使用多个条件，且这些条件可以用“&”或“|”来连接。其中，“&”用于连接任意两个条件，但两个条件必须都是真实的；而“|”只要求两个条件中的一个条件是真实的就可以。注：控制器会检测圆括号中的每个条件。

注：控制器最多只能嵌套 16 个子程序。

如果条件是真实的就会发生跳动，而条件是使用逻辑操作来检测其真实性的。这些逻辑操作符号如下：

〈 小于
〉 大于
= 等于
〈= 小于或等于

〉=大于或等于

〈〉不等于

参数: JS 子程序入口, 条件

其中子程序入口是行号或标号;

条件是使用逻辑操作的条件语句。

用途:

运动进行中	Yes
在程序中	Yes
命令行	No
适用控制器	所有型号控制器

缺省:

缺省值
缺省格式

相关命令:

“EN” 结束

举例:

JS#SQUARE, V1 〈5	如果变量 V1 小于 5 时, 跳到子程序#SQUARE
JS#LOOP, V1 〈〉 0	如果 V1 不等于 0 时, 跳到#LOOP
JS#A	跳到#A (无条件)

88) KD (二进制 83)

功能: 微分常数

说明: KD 指出控制滤波中的微分常数。滤波转换功能如下列公式所示:

$$D(z) = 4 \cdot KP + 4 \cdot KD(z-1) / z + KI \cdot z / 2(z-1)$$

参数: KD n, n, n, n, n, n, n, n 或 KDX= n

其中: n 是 0~4095.875 之间的无符号数, 分辨率为 1/8。

n=? 返回指定轴的微分常数值。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值 64
缺省格式 4.2

操作应用:

_KDn 含指定轴的微分常数值。

相关指令:

“KI”	积分常数
“KP”	比例常数

举例:

KD 100, 200, 300, 400.25	指定 A、B、C、D 轴的微分常数分别是 100, 200, 300, 400.25
KD?, ?, ?, ?	返回 A、B、C、D 轴的微分常数值
0100.00, 0200.00, 0300.00, 0400.25	

89) KI (二进制 82)

功能: 积分常数

说明: KI 命令用于设定控制回路的积分增益。它适用于如下控制等式中:

$$D(z) = 4 \cdot KP + 4 \cdot KD(z-1) / z + KI \cdot z / 2(z-1)$$

在静止时，积分将减小位置误差到 0。

参数： KI n, n, n, n, n, n, n, n 或 KIA= n

其中：n 是 0~2047.875 之间的无符号数，分辨率为 1/128。

n=? 返回指定轴的积分常数值。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	4.0

操作应用：

_KIn 读取指定轴的积分常数值。

相关指令：

“KD”	微分常数
“KP”	比例常数
“IL”	积分极限

举例：

KI 12, 14, 16, 20	指定 a, b, c, d 轴积分
KI7	只指定 a 轴
KI, , 8	只指定 c 轴
KI?, ?, ?, ?	返回 A, B, C, D 轴的积分常数值
0007, 0014, 0008, 0020	

90) KP (二进制 81)

功能： 比例常数

说明： KP 命令指定控制滤波中的比例常数。滤波转换功能如下列公式所示：

$$D(z) = 4 \cdot KP + 4 \cdot KD(z-1) / z + KI \cdot z / 2(z-1)$$

更为详细的说明，请参考操作理论部分。

参数： KP n, n, n, n, n, n, n, n 或 KPA= n

其中：n 是 0~1023.875 之间的无符号数，分辨率为 1/8。

n=? 返回指定轴的比例常数值。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	6
缺省格式	4.2

操作应用：

_KPn 读取指定轴的比例常数值。

相关命令：

“KI”	积分常数
“KD”	微分常数
“IL”	积分极限

91) KS (二进制 86)

功能: 步进电机平滑处理

说明: KS 用于设定步进电机平滑处理。此命令大多数用于全步进或半步进方式操作。KS 值越大越平滑。这个参数将通过 3KS 的采样周期来增加运动时间。在运动包络线输出时，KS 会增加单柱低通滤波。

注: 输出步时，KS 将产生延误。

参数: KS n, n, n, n, n, n, n, n 或 KSA= n
其中: n 是 .5~16 之间的正数，分辨率为 1/32。
n=? 返回指定轴的微分常数值。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	1.313
缺省格式	4.0

操作应用:

_KS_n 读取指定轴的微分常数值。

相关命令:

“MT”

定义电机类型

举例:

KS2, 4, 8

指定 a, b, c 轴

KS5

只指定 a 轴

KS, , 15

只指定 c 轴

提示: KS 命令只有使用步进电机时才有效。

92) LA

功能: 显示数组名及大小

说明: LA 命令返回存储器中的所有数组，而这些数组按字母顺序排列。

参数: 无

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	-
缺省格式	-

相关命令:

“LL”

显示程序的标号

“LS”

显示程序的指令内容

“LV”

显示变量名称及记忆值

举例:

: LA

CA[10]

LA[5]

NY[25]

93) LE (二进制 B5)

功能: 结束直线插补

说明: LE 命令用于结束直线插补。在程序中, LE 跟在 LI 命令的后面。使用 LE 后, 控制器会发出命令让电机减速到停止。LE 与 VE 可交换使用。

LE 命令适用于有坐标系选择 S 或 T 的矢量运动。要想选择坐标系, 请使用 CAS 或 CAT 命令。

参数:

n=? 返回用编码器计数来选择坐标系 S 或 T 的总矢量移动距离。要想选择坐标系, 请使用 CAS 或 CAT 命令。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	-
缺省格式	-

操作应用:

_LEn 用编码器计数读取总矢量移动距离。

相关命令:

"LI"	直线插补距离
"BG"	开始运动
"LM"	直线插补方式
"VS"	矢量速度
"VA"	矢量加速度
"VD"	矢量减速度
"PF"	位置格式

举例:

CAS	指定 S 坐标运动系
LM CD	指定 C、D 轴为直线插补方式
LI,, 100, 200	指定直线距离
LE	结束直线运动
BGS	开始运动

94) _LF*

功能: 前馈限位开关操作

说明: _LF 操作用于读取指定轴的前馈限位开关的状态。此操作按如下定义:

_LFn 其中 n 是指定轴

注: 此操作是由 CN 命令所设置的限位开关的配置情况来决定的。

对于 CN-1:

_LFn=1 当限位开关输入无效时*

_LFn=0 当限位开关输入有效时*

对于 CN1:

_LFn=0 当限位开关输入无效时*

_LFn=1 当限位开关输入有效时*

+当至少 1mA 的电流从输入电路流出, 条件有效。若输入电路配置成高电平或低电平也有效。更为详细的信息详见第三章。

举例:

MG _LFA 显示 A 轴前馈限位开关的状态

***这是操作, 不是命令。**

95) LI (二进制 B1)

功能: 直线插补距离

说明: LI a, b, c, d 命令指定了每个轴在直线插补方式 (LM) 下移动的增量式距离。LI 参数与当前轴的位置所给的距离有关。在启动程序 (BGS) 命令前, 可以写入多达 511 个程序段。在运动期间, 当缓冲区空出空间时, 可以给定附加程序段。结束直线插补命令 (LE) 必须跟在上一条 LI 命令之后, 此命令响应控制器减速到停止。用户必须在缓冲区留有足够程序段以确保连续运动。

LM? 返回 LI 程序段可利用的空间, 以便发送给缓冲区。511 返回意味着缓冲区为空, 且能发送 511 个 LI 程序段。0 意味着缓冲区满, 不能发送附加的程序段。请注意: 控制器计算矢量速度是根据 LM 模式下所指定的轴来计算的。例如, LMABC 意味着 A、B、C 轴进行直线插补。若 AS、BS、CS 为 A、B、C 轴的速度, 那么矢量速度根据公司 $VS^2=AS^2+BS^2+CS^2$ 来计算。如果 LI 命令只指出 A、B 轴, 那么 C 轴的速度将一直用于矢量计算。控制器总是使用 LM 指出的轴来计算速度而不是 LI。参数 n 是可选择的, 它用于定义附加在运动程序段的矢量速度。

LI 命令适用于选择坐标系 S 或 T。为了选择坐标系, 用户可以使用 CAS 或 CAT 命令。

参数: LI n, n, n, n, n, n, n, n, n <0> p 或 LIA= n

其中: n 是 -8, 388, 607~8, 388, 607 之间的带符号整数, 表示增量移动距离。

o 指定执行直线程序段的矢量速度。对于伺服电机来说, o 是 0~12, 000, 000 之间的无符号偶整数; 而对步进电机而言, o 是 0~3, 000, 000 之间的无符号偶整数。

p 指定结束直线程序段时所达到的矢量速度。基于矢量加、减速比, o 是 0~8, 000, 000 之间的无符号偶整数。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	-
缺省格式	-

相关命令:

“LE”	结束直线插补
“BG”	开始

“LM”	直线插补方式
“CS”	清除顺序
“VS”	矢量速度
“VA”	矢量加速度
“VD”	矢量减速度

举例：

LM ABC	指定直线插补方式
LI 1000, 2000, 3000	指定直线插补距离
LE	结束直线插补
BGS	开始启动

96) LL

功能：显示程序的标号

说明：LL 命令返回存储器中关于所有程序标号的一揽表，而这揽表将按字母顺序排列。

参数：无

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	-

相关命令：

“LA”	显示数组名及大小
“LS”	显示程序的指令内容
“LV”	显示变量名称及记忆值

举例：

```
: LL
#FIVE
#FOUR
#ONE
#THREE
#TWO
```

97) LM (二进制 B0)

功能：直线插补方式

说明：LM 命令用于指定直线插补方式及直线插补方式中的轴。在直线插补方式中可以使用 1~8 轴。LI 命令指定直线插补距离，LE 命令用于结束直线插补。只要缓冲区有空间给附加程序段，控制器就可以给出几个 LI 命令。一旦程序执行 LM 命令后，除非使用 VM 命令，否则不需要再用 LM 指令。

值得注意的是控制器计算矢量速度是建立在 LM 方式下所指定轴的基础上。例如，LM ABC 是指在直线插补方式中指定 A, B, C 轴，而轴的速度将根据 $VS^2=AS^2+BS^2+CS^2$ 公式来计算，其中 AS, BS, CS 表示 A, B, C 轴的速度。在这个例子中，如果 LI 命令只定义了 A, B 轴，在矢量速度计算中要使用 C 轴速度，控制器要求用 LM 命令去计算，而 LI 命令不行。

LM 命令适用于选择坐标系 S 或 T。为了选择坐标系统，用户可以使用 CAS 或 CAT 命令。

参数： LM nnnnnnnnnnn

其中：n 是 A, B, C, D, E, F, G, H 中的一轴或他们的任意组合。

n=? 用附加的 LI 命令返回程序缓冲区中可用的空间数。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	-

操作应用：

_LMn 用 'n' 坐标系，S 或 T 读取程序缓冲区中可用的空间数。

相关命令：

"LE"	结束直线插补
"LI"	直线插补距离
"CS"	清除运动序列
"VS"	矢量速度
"VA"	矢量加速度
"VD"	矢量减速度
"AV"	等待某矢量位置到达

举例：

LM ABCD	指定直线插补方式
VS 10000: VA100000; VD1000000	指定矢量速度、加速度、减速度
LI100, 200, 300, 400	指定直线距离
LI200, 300, 400, 500	指定直线距离
LE; BGS	结束直线插补后再开始运动

98) _LR*

功能： 反向限位开关操作

说明： _LR 操作用于读取指定轴的反向限位开关的状态。此操作按如下定义：

_LRn 其中 n 是指定轴

注： 此操作是由 CN 命令所设置的限位开关的配置情况来决定的。

对于 CN-1:

_LRn=1 当限位开关输入无效时*

_LRn=0 当限位开关输入有效时*

对于 CN1:

_LRn=0 当限位开关输入无效时*

_LRn=1 当限位开关输入有效时*

+当至少 1mA 的电流从输入电路流出，条件有效。若输入电路配置成高电平或低电平也有效。

举例：

MG - LRA 显示 A 轴反向限位开关的状态

***这是操作，不是命令。**

99) LS

功能：显示程序的指令内容

说明：LS 命令返回存储器中一揽程序表。

参数：LS n, m

其中：n 和 m 是 0~999 之间的有效数字或标号。n 是所列程序的第一行，m 是最后一行。

n 是 0~999 之间的整数或标号。n 用于指定所列程序的第一行。

m 是 0~999 之间的整数或标号。m 用于指定所列程序的最后一行。

用途：

运动进行中	Yes
在程序中	No
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0, 最后一行
缺省格式	-

相关命令：

“LA”	显示数组名及大小
“LL”	显示程序的标号
“LV”	显示变量名称及记忆值

举例：

: LS#A, 6	显示#A 至第 6 行的程序内容
002#A	
003 PR 500	
004 BGA	
005 AM	
006 WT 200	

提示：在使用 LS 命令前，用 Cntrl+Q 退出编辑模式。

100) LV

功能：显示变量名称及记忆值

说明：LV 命令返回存储器中关于所有程序变量的一揽表，而此表是按字母顺序排列。

参数：无

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	-

相关命令：

“LA”	显示数组名及大小
“LS”	显示程序的指令内容

“LL”

显示程序的标号

举例：

```
: LV
APPLE=60.0000
BOY=25.0000
ZEBRA=37.0000
```

101) LZ (二进制 E7)

功能：将有效值之前的零点移除

说明：LZ 命令用于格式化从问询命令、问询变量及数组中返回的值。通过 LZ 命令，将移除有效值之前的零点。

参数：LZ n

其中：n=1 移除有效值之前的零点。

n=0 不能移除有效值之前的零点。

n=? 返回 LZ 状态。为 ‘0’ 时，不能移除有效值之前的零点；为 ‘1’ 时，可以移除。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	1
缺省格式	-

操作应用：

—LZ 读取 LZ 状态。为 ‘0’ 时，不能移除有效值之前的零点；为 ‘1’ 时，可以移除。

举例：

LZ0	不能移除有效值之前的零点
TPA	查询 A 轴当前位置值
0000021645.0000	控制器返回值
VAR1=	查询 “VAR1” 变量值（优先设成 10）
0000000010.0000	控制器返回变量值
LZ1	移除有效值之前的零点
TPA	读取 A 轴当前位置
21645.0000	控制器返回值
VAR1=	查询 “VAR1” 变量值（优先设成 10）
10.0000	控制器返回变量值

102) MB

功能：Modbus

说明：MB 命令使用 Modbus 协议的前两级与 I/O 装置进行通信。

根据每个功能代码的不同，此命令的格式也有所变化。代码-1 表明使用了 Modbus 的第一级，其它的代码是 DMC-2100 支持的第二级的 10 个主要功能代码。

功能代码	定义
01	读置位状态（读位）
02	读输入状态（读位）

03	读保持寄存器（读字）
04	读输入寄存器（读字）
05	强制输出置位（写 1 位）
06	预置寄存器（写 1 字）
07	读额外状态（读错误码）
15	强制多输出点置位（写多位）
16	预置多寄存器（写字）
17	报告从 ID

注 1: 所有命令格式都有地址，在这是从地址。从地址的指定或缺省是由装置端口数来决定的。

注 2: 所有的格式都含 h 参数，表明连接端口 A~F.

参数:

MBh=-1, len, array[]

其中: len 是字节数

array[] 是带数据的数组名

MBh=addr, 1, m, n, array[]

其中: m 是开始的位数

n 是位号

array[] 表明持续第一个元素的结果

MBh=addr, 2, m, n, array[]

其中: m 是开始的位数

n 是位号

array[] 表明持续第一个元素的结果

MBh=addr, 3, m, n, array[]

其中: m 是开始的寄存器数

n 是寄存器号

array[] 将保持响应

MBh=addr, 4, m, n, array[]

其中: m 是开始的寄存器数

n 是寄存器号

array[] 将保持响应

MBh=addr, 5, m, n

其中: m 是开始的位数

n 为 0 或 1, 表明置位设成关或开

MBh=addr, 6, m, n

其中: m 是寄存器号

n 是 16 位值

MBh=addr, 7, array[]

其中: array[] 是返回数据所存放的位置 (1 字节/元素)

MBh=addr, 15, m, n, array[]

其中: m 是开始的位数

n 是位号

array[] 读取数据 (1 字节/元素)

MBh=addr, 16, m, n, array[]

其中：m 是开始的寄存器数

n 是寄存器号

array[] 读取数据（16 位字/元素）

MBh=addr, 17, array[]

其中：array[] 是返回数据所存放的位置

用途：

运动进行中 Yes

在程序中 Yes

命令行 Yes

适用控制器 DMC-2100, DMC-2200

缺省：

缺省值 -

缺省格式 -

103) MC (二进制 C9)

功能： 定位完毕

说明： MC 命令是一个条件启动，用于控制事件时序。此命令将使后续命令的执行挂起，直到该轴当前运动完成且编码器到达或超过指定位置。用 MC 命令可以指定这些轴的任意组合。例如，MC AB 是等待 A, B 轴都完成定位。若 MC 命令不带参数则表明完成所有轴的定位。如果编码器在规定的时间内没有在指定的位置，那么 TW 命令会设置时间溢出并发出错误提示。如果发生时间溢出这种情况，将会清除条件启动，且停止代码被设成 99，同时一个应用程序将会跳到指定的标号。

当使用步进方式时，控制器将使后续命令的执行挂起，直到此控制器在命令位置形成相同的步数。而实际的步数可以通过使用查询命令 TD 来监控。注意：当使用步进电机时，通常推荐使用 MC 命令，因为步进电机平滑处理功能（KS）会导致步进脉冲延误。

另外，MC 命令只有产生所有的步数后才能满足。

参数： MC nnnnnnnn

其中：n 是 A, B, C, D, E, F, G, H 或者所要指定轴的任意组合。

若不带参数，表明完成所有轴的运动。

用途：

运动进行中 Yes

在程序中 Yes

命令行 Yes

适用控制器 所有型号控制器

缺省：

缺省值 -

缺省格式 -

相关命令：

“BG”

启动程序

“AM”

等待运动完成

“TW”

Time-out 事件

举例：

#MOVE

程序名

PR2000, 4000

设定 A、B 轴的相对位置分别是 2000、4000

BG AB

A、B 轴开始运动

MC AB

在 T 坐标系中等待运动完成后

MG “DONE”; TP

显示信息

EN

程序结束

提示： MC 命令可用于检验实际运动是否已完成。

104) MF (二进制 CB)

功能: 正向移动到某一个位置

说明: MF 命令是一个条件启动, 用于控制事件时序。此命令将使后续命令执行挂起, 直到指定电机正向移动及越过指定的位置*。此命令值的单位是 4 倍频计数单位。1 次只能指定 1 轴。MF 命令只要求带编码器, 而不需要伺服控制。

*使用步进电机时, 当步进位置 (由输出缓冲区来决定) 越过指定的正向运动位置时, 此条件可以满足。更为详细的信息请参考用户手册“步进电机操作”第 6 章节。

参数: MF n, n, n, n, n, n, n, n, n or MFA=n

其中: n 是-2147483648~2147483647 之间的十进制带符号整数。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	-
缺省格式	-

相关命令:

“AD”	等待某位置到达
“AP”	等待某绝对位置到达

举例:

#TEST	程序名
DP0	定义零点
JG 1000	Jog 模式 (速度为 1000 计数/秒)
BG A	启动程序
MF 2000	移动到 2000 这个位置后
V1= _TPA	分配变量 V1 为 A 轴实际位置值
MG “Position is”, V1	显示信息
ST	停止
EN	程序结束

提示: MF 命令的精度是在 2msec 内出现的计数值, 用 2msec 乘以速度来得到最大误差。MF 检测绝对位置。当外部设备独立驱动指定的电机时也可以使用此命令。

105) MG

功能: 显示信息

说明: MG 命令发送数据给总线。此命令可用于改变操作、发送提示或返回变量值。

参数: MG “m”, { ^ n}, V {Fm.n or \$m,n} {N} {Pn}

其中: “m” 是文本信息, 包括字母、数字、符号或 <ctrl> G (多达 72 个字符数)。

{ ^ n} 通过 n 值指定 ASCII 码字符数。

{U} 是 USB 通信口

{Ex} 是 Ethernet 及 ‘x’ 指定的 Ethernet 端口 (A, B, C, D, E, F 或 H)

V 是一个变量名或数组元素, 它用于下列格式:

{Fm.n} 用十进制格式显示变量, 其中 m 在十进制左边, n 在右边。

{\$m.n} 用十六进制格式显示变量, 其中 m 在十进制左边, n 在右边。

{Sn} 显示长度为 n 的字符串变量, 其中 n 是 1~6 之间的数。

{N} 强制换行

{Pn} 当 n 为 0 或 1 时，指出输出到 P1 主串口或 P2 辅助串口。

注：可以使用多个文本、变量或 ASCII 码字符，但每个之间必须用逗号隔开。

注：此参数不重要。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	变量格式

举例：

第一种：信息命令显示 ASCII 字符串

MG “Good Morning” 显示字符串

第二种：信息命令显示变量或数组

MG “The Answer is”, Total {F4.2}

显示格式为小数点前 4 位后 2 位的 TOTAL 变量的内容

第三种：信息命令给通信口发送 ASCII 字符。

MG { ^ 13}, { ^ 10}, { ^ 48}, { ^ 055} 显示回车及 0~7 之间的字符。

106) MO (二进制 A9)

功能：电机关断

说明：MO 命令用于关闭控制系统，而控制器将继续监控电机位置。要想打开电机，必须使用 SH 命令来激活伺服状态。

参数：MO nnnnnnnnnn

其中：n 是 A, B, C, D, E, F, G, H 或者所要指定轴的任意组合。

若不带参数，指定所有轴。

用途：

运动进行中	No
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	1.0

操作应用：

_MO_n 读取指定轴的电机状态。

相关命令：

“SH” 激活伺服状态 (servo on)

举例：

MO	关断所有电机
MOA	关断 A 轴电机，其它电机不变
MOB	关断 B 轴电机，其它电机不变
MOCA	关断 A、C 轴电机，其它电机不变
SH	激活所有电机

B0b= _MOA 设置 A 轴伺服状态为变量 B0b 值

B0b= 返回 B0b 值。若是电机关断模式此值为 1，若是伺服模式此值为

提示：MO 命令常用于手动电机位置方式。若想激活伺服状态，用 SH 命令。

107) MR (二进制 CC)

功能：反向移动到某一个位置

说明：MR 命令是一个条件启动，用于控制事件时序。此命令将使后续命令执行挂起，直到指定电机反向移动及越过指定的位置*。此命令值的单位是 4 倍频计数单位。1 次只能指定 1 轴。MR 命令只要求带编码器，而不需要伺服控制。

*使用步进电机时，当步进位置（由输出缓冲区来决定）越过指定的反向运动位置时，此条件可以满足。更为详细的信息请参考用户手册“步进电机操作”第 6 章节。

参数：MR n, n, n, n, n, n, n, n or MRA=n

其中：n 是-2147483648~2147483647 之间的十进制带符号整数。

用途：

运动进行中	No
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值
缺省格式

相关命令：

“AD”

等待某位置到达

“AP”

等待某绝对位置到达

举例：

#TEST	程序 B
DP0	定义零点
JG -1000	Jog 模式（速度为 1000 计数/秒）
BG A	启动程序
MR -3000	在越过位置-3000 后
V1= _TPA	分配变量 V1 为 A 位置
MG “Position is”, V1=ST	显示信息，停止
EN	程序结束

提示：MR 命令的精度是在 2msec 内出现的计数值，用 2msec 乘以速度来得到最大误差。MR 检测绝对位置。当外部设备独立驱动指定的电机时也可以使用此命令。

108) MT

功能：指定电机类型

说明：MT 命令用于选择电机类型及驱动信号的极性。电机包括伺服电机（要求电压为+/-10V 之间）及步进电机（脉冲+方向形）。对于伺服电机，极性翻转成模拟信号，而对于步进电机来说，极性翻转为脉冲逻辑电平。

参数：MT n, n, n, n, n, n, n, n or MTA=n

其中：n=1	指定伺服电机
n=-1	指定反极性伺服电机
n=-2	指定有效高步进脉冲的步进电机
n=2	指定有效低步进脉冲的步进电机
n=-2.5	指定反方向有效高步进脉冲的步进电机

n=2.5 指定反方向有效低步进脉冲的步进电机
 n=? 返回指定轴的电机类型值

用途:		缺省:	
运动进行中	No	缺省值	1, 1, 1, 1
在程序中	Yes	缺省格式	1
命令行	Yes		
适用控制器	所有型号控制器		

操作应用:

_MTn 读取指定轴的电机类型值。

相关命令:

“CE” 设定编码器类型

举例:

MT1, -1, 2, 2 设定 a 为伺服, b 为反极性伺服, c 和 d 为步进
 MT? , ? 查询电机类型
 V= _MTA 分配电机类型给变量

109) NB

功能: 凹陷滤波带宽

说明: NB 命令用于设置凹陷滤波极点的实部分。

参数: NB n, n, n, n, n, n, n, n or NBA=n

其中: n 是 0Hz~1/ (16° TM) 之间的数。

用途:		缺省:	
运动进行中	Yes	缺省值	0.5
在程序中	Yes	缺省格式	
命令行	Yes		
适用控制器	所有型号控制器		

操作应用:

_NBn 读取指定轴凹陷滤波的带宽值。

相关命令:

“NF” 凹陷滤波频率
 “NZ” 凹陷滤波零点

举例:

_NBA=10 设置凹陷滤波极点的实部分为 10/2Hz
 NOTCH=_NBA 设置变量” NOTCH” 为 A 轴凹陷滤波带宽值

110) NF

功能: 凹陷滤波频率

说明: NF 命令用于设置用 PID 进行补偿的凹陷滤波频率。

参数: NF n, n, n, n, n, n, n, n, n or NFA=n

其中: n 是 1Hz~1/ (4° TM) 之间的数, TM 是更新率。(缺省时 TM 为 1msec)

n=? 返回指定轴的凹陷滤波频率值。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	0
缺省格式	

操作应用:

_NFn 读取指定轴的凹陷滤波频率。

相关命令:

“NF”	凹陷滤波带宽
“NZ”	凹陷滤波零点

举例:

NF, 20	设置 B 轴的凹陷滤波频率为 20Hz
--------	---------------------

111) NO (也接受撇号 (‘))

功能: 作批注. 无作用

说明: 在程序中使用 NO 或撇号 (‘), 无作用, 但可以当作注释用于程序中, 它有利于使用程序中的文件。

参数: NO m

其中: m 是任意字母及数字组

NO 命令后面可以跟多达 77 个字符。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	
缺省格式	

举例:

#A	程序 A
NO	无操作
NO This Program	无操作
NO Does Absolutely	无操作
NO Nothing	无操作
EN	程序结束

112) NZ

功能: 凹陷滤波零点

说明: NZ 命令用于设置凹陷滤波零点的实部。

参数: NZ n, n, n, n, n, n, n, n or NZA=n

其中: n 是 1Hz~1/ (16° TM) 之间的数。

n=? 返回指定轴凹陷滤波零点值。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	0.5
缺省格式	

操作应用:

_NZn 读取指定轴凹陷滤波零点值。

相关命令:

"NF"	凹陷滤波频率
"NB"	凹陷滤波带宽

举例:

NZA=10 设置凹陷滤波极点的实部为 10/2Hz

113) OB (二进制 E9)

功能: 位输出

说明: OB 命令根据表达式的结果来定义 0 还是 1。任何非零值在输出时为 1。

参数: OB n, expression

其中: n 表示位输出

expression 是任意有效的逻辑表达式、变量或数组元素。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	
缺省格式	

举例:

OB 1, POS=1	若 POS1 非零, 那么位 1 为高电平 若 POS1 为零, 那么位 1 为低电平
OB 2, @IN[1] & @IN[2]	若输入 1 和输入 2 都高, 那么输入 2 设成高电平
OB 3, COUNT[1]	若元素 1 在数组中为零, 那么清除位 3
OB N, COUNT[1]	若元素 1 在数组中为零, 那么清除位 N

114) OC

功能: 输出比较

说明: OC 命令根据主编码器之一的位置产生输出脉冲。在输出比较信号 (在 ICM-1900 及 ICM-2900 上所标的 CMP) 时, 可提供大约 600 毫微秒/周期的低沿脉冲。

使用此命令, 不能用于步进电机及辅助编码器。

注: OC 命令要求主、辅编码器的配置完全相同 (见命令 CE) 例如: CE0, CE5, CE10, CE15。

参数: OC x=m, n

其中: x=A、B、C、D、E、F、G、H 指定用于哪个编码器输入。

m=首脉冲绝对位置。范围是 -2×10^9 ~ 2×10^9 之间的整数。

n =脉冲之间的增量式距离。范围是-65535~65535 之间的整数。

注：OCx=0 没有圆弧比较功能。

参数 n 将指出输出比较时所运动的方向。当移动到反方向时，脉冲在增量式距离 65536- $|n|$ 位置上发生输出比较，而 $|n|$ 是 n 的绝对值。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	-

操作应用：

$_OCx$ 读取 OC 功能的状态。

$_OCx=0$ ：具备 OC 功能，但不产生任何脉冲

$_OCx=1$ ：OC 函数无效或已产生首输出脉冲

举例：

OCA=300, 100 选择 A 编码器作为位置传感器，首脉冲为 300，后续脉冲为 400，500 ……

115) OE (二进制 8D)

功能：位置错误关断电机

说明：如果位置误差超出 ER 命令所指定的极限时，OE 命令会导致控制器关断电机，此时可以采用 AB 命令或急停输入去改变现状。

如果检测出位置误差发生在一个轴上，那么除了这个轴关断电机外，其它轴仍可继续运行。如果位置误差发生在像 VM、LM、CM 这种联动模式中，那么所有的轴都会关断电机。

参数：OE $n, n, n, n, n, n, n, n, n$ or OEA= n

其中： $n=0$ 没有位置错误关断电机功能
 $n=1$ 位置错误关断电机功能

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	---

操作应用：

$_OEn$ 读取指定轴位置错误关断电机功能的状态。为 0 时，没有这种功能，当为 1 时有这种功能。

相关命令：

“Error! Not a valid result for table”	急停
“ER”	最大误差极限
“SH”	激活伺服状态
#POSERR	错误处理子程序

举例：

OE1, 1, 1, 1	位置错误时，所有轴都会关断电机功能
OE0	位置错误时，除 A 轴外其它轴都会关断电机功能
OE, , 1, 1	位置错误时，C、D 轴关断电机，其它两轴继续运行
OE1, 0, 1, 0	位置错误时，A、C 轴关断电机，其它两轴继续运行

提示：OE 命令可以避免系统由于过多误差引起的损坏。

116) OF(二进制 99)

功能：零点补偿

说明：当电机命令输出或返回先前的设置值时，OF 命令会设置一个偏压，从而抵制重力或在放大器中设置零点补偿。

参数：OF n, n, n, n, n, n, n, n or OFA=n

其中：n 是-9.998~9.998 之间的带符号数，分辨率为 0.0003。

n=? 返回指定轴的零点补偿。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	1.0

操作应用：

_OFn 读取指定轴零点补偿。

举例：

OF1, -2, 3, 5	设置 A、B、C、D 轴零点补偿分别为 1、-2、3、5
OF-3	设置 A 轴零点补偿为-3，其它轴不变
OF, 0	设置 B 轴零点补偿为 0，其它轴不变
OF?, ?, ?, ?	返回零点补偿
-3.0000, 0.0000, 3.0000, 5.0000	
OF?	返回 A 轴零点补偿
-3.0000	
OF, ?	返回 B 轴零点补偿
0.0000	

117) OP(二进制 E8)

功能：输出通信口

说明：OP 命令发送数据给控制器的输出通信口。您可以使用输出通信口去控制外部开关和继电器。

参数：OP m, a, b, c, d

其中：m 是 0~65535 之间的十进制整数或\$0000~\$FFFF 之间的十六进制整数。(0~255 用于 4 轴或 4 轴以下)。它还可以用十进制来表示通用输出位数。输出 1~8 用于控制 1~4 轴控制器，输出 1~16 用于控制 5 轴或 5 轴以上的控制器。

a, b, c, d 代表以 16 位为一组的扩展 I/O, (值从 0~65535)。若作为输入配置的 I/O 点给参数，此参数将被忽略。具体输出情况的设置，详见下表：

参数	块	位数	说明
----	---	----	----

m	0	1-8	通用输出（1-4 轴控制器）
	0, 1	1-16	通用输出（5-8 轴控制器）
a	2, 3	17-32	扩展 I/O
b	4, 5	33-48	扩展 I/O
c	6, 7	49-64	扩展 I/O
d	8, 9	65-80	扩展 I/O

n=? 返回参数值，其中 n 是以上任意的参数。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	3.0

操作应用：

_OP0 读取第一个参数值，m

_OP1 读取第一个参数值，a

_OP2 读取第一个参数值，b

_OP3 读取第一个参数值，c

_OP4 读取第一个参数值，d

相关命令：

“SB”	设置输出位
“CB”	清除输出位
“OB”	输出字节

例子：

OP0	清除输出通信口—所有位
OP\$85	设置输出 1, 3, 8; 清除其它的输出
MG_OP0	返回第一个参数 “m”
MG_OP1	返回第二个参数 “a”

118) PA(二进制 A6)

功能：绝对位置

说明：PA 命令为每个轴设置最终位置，而这个位置是针对于坐标零点来设置的。

参数：PA n, n, n, n, n, n, n, n or PAA=n

其中：n 是-2147483647~2147483648 之间的十进制带符号整数。此值是用编码器计数作单位。

n=? 在运动停止时返回命令位置。

用途：

运动进行中	No
-------	----

缺省：

缺省值	-
-----	---

在程序中	Yes	缺省格式	位置格式
命令行	Yes		
适用控制器	所有型号控制器		

操作应用:

_PAn 运动停止时, 读取上一命令位置。

相关命令:

“PR”	相对位置
“SP”	速度
“AC”	加速度
“DC”	减速度
“BG”	开始
“PF”	位置格式

举例:

: PA400, -600, 500, 200	A、B、C、D 轴将分别走 400, -600, 500, 200 个计数时的位置
BG; AM	执行运动, 等待运动完成
: PA? , ? , ? , ?	运动完成后, 返回当前命令位置
400, -600, 500, 200	
: BG	开始运动
: PA700	在下一个运动时, A 轴将走到 700, 而
: BG	B、C、D 轴将会按照先前设置的相对距离运动, 若进行的是 PR 运动; 如果进行的运动是 PA 运动时, B、C、D 轴将会停下来

119) PF

功能: 位置格式

说明: PF 命令允许用户去格式化例如 TP 返回的那些位置值。此命令可以选择整数位数及小数位数。其另加的符号位及小数点位也将加到总位数上。如果 PF 是负数时, 此时格式为十六进制数, 且在字符前加一个\$符号。十六进制数用 2' s 来表示。

如果它的数字超出了格式, 那么这个数字将会按照最大可能的正数或负数来显示。(例如 999.99, -999, \$8000 或\$7FF)。

PF 命令用于格式化从下列命令中返回的值:

BL?	LE?
DE?	PA?
DP?	PR?
EM?	TN?
FL?	VE?
IP?	TE
TP	

参数: PF m, n

其中: m 是-8~10 之间的整数, 它置于小数点前, 若它是个负数, 则用十六进制数表示。

n 是 0~4 之间的整数, 它置于小数点之后。

n=? 返回 m 值。

用途:		缺省:	
运动进行中	Yes	缺省值	10.0
在程序中	Yes	缺省格式	10.0
命令行	Yes		
适用控制器	所有型号控制器		

操作应用:

_PF 读取 'm' 位置格式参数值。

举例:

: TPX	响应 X 位置
0000000000	缺省格式
: PF5.2	改变格式, 整数为 5, 小数为 2
: TPX	响应位置
00021.00	
: PF-5.2	重新改变格式为十六进制数
: TPX	响应位置
\$00015.00	用十六进制表示

120) PL(二进制 87)

功能: 低通滤波的 PID 补偿

说明: PL 命令增加了低通滤波的 PID 补偿功能。这个数字转移功能可以用 $(1-P)/(z-P)$ 来表示, 与它相等的连续滤波用 $A/(S+A)$ 表示, 其中 A 是滤波切断频率: $A = (1/T) \ln(1/p)$ rad/sec, 而 T 是采样时间。

参数: PL n, n, n, n, n, n, n, n, n or PLA=n

其中: n 是 0~0.9999 之间的正数。

n=? 返回指定轴低通滤波的 PID 补偿值。

用途:		缺省:	
运动进行中	Yes	缺省值	0.0
在程序中	Yes	缺省格式	3.0
命令行	Yes		
适用控制器	所有型号控制器		

操作应用:

_PLn 读取指定轴低通滤波的 PID 补偿值。

相关命令:

"KD"	微分常数
"KP"	比例常数
"KI"	积分常数

举例:

PL.95,.9,.8,.822	设置 A、B、C、D 轴的 PID 补偿值分别是 0.95, 0.9, 0.8, 0.822
PL?, ?, ?, ?	返回所有轴的 PID 补偿值
0.9527, 0.8997, 0.7994, 0.8244	
PL?	只返回 A 轴的 PID 补偿值
0.9527	

PL? 只返回 B 轴的 PID 补偿值
0.8997

121) PR(二进制 A7)

功能: 相对位置

说明: PR 命令设置增量式距离及下一次运动的方向. 这个运动与当前位置有关.

参数: PR n, n, n, n, n, n, n, n or PRA=n

其中: n 是-2147483648~2147483647 之间的十进制带符号整数. 此值是用编码器计数作单位。

n=? 返回指定轴的当前增量式距离。

用途:

运动进行中	No
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	0
缺省格式	位置格式

操作应用:

_PRn 读取指定轴的当前增量式距离。

相关命令:

“PA”	绝对位置
“SP”	速度
“AC”	加速度
“DC”	减速度
“BG”	开始
“PF”	位置格式
“IP”	增量位置

举例:

: PR100, 200, 300, 400	在执行下一个运动时, A、B、C、D 轴将分别走
: BG	100, 200, 300, 400 个计数
: PA? , ? , ?	返回相对距离
0000000100, 0000000200, 0000000300	
: PR500	设置 A 轴相对距离为 500
: BG	在下一个运动中, A 轴将走 500 个计数, 而 B 轴将移动到它
	先前设置的相对距离的那个位置

122) QD

功能: 下载数组

说明: QD 命令将数组数据从主计算机传送到控制器. QD array [] , start, end 要求指定第一个数组元素及最后一个数组元素的数组名. 各数组元素之间用逗号或<CR><LF>隔开, 用<control>Z, <control>Q, <control>D 或 \ 结束下载.

参数: QD array [] , start, end

其中: array [] 是有效的数组名, start 是第一个数组元素(缺省值为 0), end 是最后一个数组元素(缺省值为最后一个元素).

用途:

缺省:

运动进行中	No	缺省值	0
在程序中	Yes	缺省格式	位置格式
命令行	Yes		
适用控制器	所有型号控制器		

相关命令:

“QU” 上传数组

提示:

使用 GALIL terminal 软件, 此命令可用于下列方式中:

1. 设置时间溢出为 0
2. 发送命令 QD
3. 使用发送文件命令去发送数据文件或从 terminal 进入手动操作, 用符号“\”来结束上传数据。

123) QR

功能: 数据记录

说明: QR 命令可以使控制器返回关于控制器状态的记录信息. 此信息是由 4 字节的启头信息及命令参数指定的信息块组成. 具体的情况详见用户手册第四章。

参数: QR nnnnnnnnnn

其中: n 是 A, B, C, D, E, F, G, H, S, T, I 或者所要指定轴的任意组合或程序或 I/O 状态.
S 和 T 代表 S 和 T 坐标运动平台.

I 代表 I/O 状态

用户手册第四章提供了如何定义数据记录信息.

用途:

缺省:

运动进行中	Yes	缺省值	-
在程序中	Yes	缺省格式	-
命令行	Yes		
适用控制器	所有型号控制器		

相关命令:

“QZ” 返回 DMC/数据记录信息

注: GALIL 中的 Terminal 将不会显示 QR 命令的结果, 因为这个结果是个二进制格式。

124) QU

功能: 上传数组

说明: QU 命令将数组数据从控制器传送到主计算机. QU 要求第一个数组元素及最后一个数组元素所指定的数组名. 上传数组跟在<control>Z 后面作为文本标号的结束。

参数: QU array [], start, end, delim

其中: “array []” 是有效的数组名

“start” 是第一个数组元素(缺省值为 0)

“end” 是最后一个数组元素(缺省值为最后一个元素)

“delim” 用于划定数组元素界限的字符. 如果 delim 是 1, 那么数组元素将被逗号隔开, 否则, 数组元素将会被回车键隔开。

用途:

缺省:

运动进行中	No	缺省值	0
在程序中	Yes	缺省格式	位置格式

命令行	Yes
适用控制器	所有型号控制器
相关命令:	
“QD”	下载数组

125) QZ

功能: 返回 DMA/数据记录信息

说明: QZ 命令是查询命令, 此命令返回关于 DMA 转移 (DMC-1700)、第二 FIFO (DMC-1600、DMC-1700、DMC-1800) 或数据记录 (DMC-1200、DMC-2000、DMC-2100) 的信息。控制器对于这个命令所做出的响应是返回用逗号隔开的 4 个整数。这 4 领域按如下所示表示:

第一领域返回轴数

第二领域返回一般状态下传送的字节数

第三领域返回坐标运动状态下传送的字节数

第四领域返回轴指定信息传送的字节数

参数: QZ

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	---
缺省格式	

相关命令:

“DR”	DMA 上传率
“QR”	数据记录

126) RA

功能: 记录数组

说明: RA 命令选择 1~8 个数组进行自动数据捕获, 而所选择的数组必须由 DM 命令来定义。RD 命令可以指定捕获的数据, RC 命令可以指定时间间隔。

参数: RA n [], m [], o [], p [] RA n [], m [], o [], p [] q [], r [], s [], t []

其中: n, m, o, p 是由 DM 命令定义的数组。[] 内为空。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	-
缺省格式	-

相关命令:

“DM”	定义数组
“RD”	记录数据
“RC”	记录间隔

举例:

#Record	标号
DM POS [100]	定义数组

RA POS []	定义记录模式
RD _TPA	定义记录的数据类型
RC1	在 2 微秒间隔时开始记录
PR 1000; BG	开始运动
EN	结束程序

提示：记录数组模式用于记录电机在运动时的实时位置。数据是在后台自动捕获，不会中断程序。记录数组模式也用于示教或学习运动轨迹。

127) RC

功能：记录

说明：RC 命令用于自动记录数组模式（RA）中开始记录。RC0 表明停止记录。

参数：RC n, m

其中：n 是 1~8 的整数, 在记录时指定 2^n 采样。RC0 时停止记录。

m 是可选参数, 指定被记录的记录号。若未指定参数 m, 可以使用 DM 命令数。若 m 是负数时导致圆弧记录越过数组地址 0 到 m-1。下一次记录的数组元素的地址可以用 _RC 查询。

n=? 返回记录情况, 若记录, 为 1; 不记录, 为 0。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	-

操作应用：

_RC 读取记录情况, 若记录, 为 1; 不记录, 为 0。

相关命令：

“DM”	定义数组
“RD”	记录数据
“QZ”	记录数组模式

举例：

#Record	记录
DM Torque [1000]	定义数组
RA Torque []	指定记录模式
RD _TTA	指定数据类型
RC2	开始记录, 在记录时设成 4 微秒
JG 1000; BG	开始运动
#A; JP#A, _RC=1	循环直到完成
MG “DONE RECORDING”	显示信息
EN	结束程序

128) RD

功能：记录数据

说明：RD 命令指定记录数组（RA）模式中捕获的数据类型，这些命令类型如下表所示：

_AFn	模拟输入值（+32767~-32768）。
_DEn	第二编码器
_TPn	位置
_TEn	位置误差
_SHn	命令位置
_RLn	锁存位置
_TI	输入
_OP	输出
_TSn	开关，只有 0~4 位有效
_SCn	停止代码
_TTn	响应转矩(注：转矩记录值的范围是+/-32767,其中 0 为 0 转矩,-32767 为-10V 命令输出, +32767 为+10V 命令输出)

其中：‘n’是指定轴,A…H。

参数：RDm1, m2, m3, m4, m5, m6, m7, m8

其中：参数是用记录数组特征捕获的数据类型。此命令非常重要，每个数据类型与 RA 命令所指定的数组保持一致。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	-

操作应用：

_RD 读取记录时下一个数组元素的地址。

相关命令：

“DM”	定义数组
“RC	记录间隔
“QZ”	记录数组模式

举例：

DM ERRORA [50], ERRORB [50]	定义数组
RA ERRORA [], ERRORB []	指定记录模式
RD _TEA, _TEBS	指定数据类型
RC1	开始记录
PR 1000; BG	开始运动

129) RE

功能：从处理错误返回主程序

说明：RE 命令用于结束位置错误处理子程序或极限开关处理子程序，其中错误处理子程序是以#POSERR 启头的，而极限开关处理子程序是以#LIMSWI 启头的。在结束这些子程序后，RE 命令会返回到主程序，此时为了避免重新进入子程序中，错误及限位开关这些条件不再存在。如果程序在等待条件启动发生且使用了 RE1 命令，那么在返回程序时会恢复先前的错误中断这个条件启动。若用了 RE0，那么将会清除条件启动。为了避免在中断时返回到主程序，可使用 ZS 命令进行程序堆栈。

参数：RE n

其中：n=0 清除中断条件启动

 n=1 恢复条件启动

若不带参数，则清除中断条件启动。

用途：

运动进行中	No	缺省值	-
在程序中	Yes	缺省格式	-
命令行	No		
适用控制器	所有型号控制器		

缺省：

相关命令：

#POSERR	错误处理子程序
#LIMSWI	限位处理子程序

举例：

#A; JP#A; EN	主程序标号
#POSERR	开始错误处理子程序
MG "ERROR"	显示信息
SB1	设置输出位 1
RE	返回主程序且清除条件启动

提示：应用程序必须执行#LIMSWI 及#POSERR 子程序命令。

130) RI

功能：从中断点返回主程序

说明：RI 命令用于结束以#ININT 启头的中断子程序，且返回到主程序中。RI 命令可以重新使能输入中断。如果程序在等待条件启动例如 WT 命令时发生中断，RI1 将在返回程序中恢复条件启动，而 RI0 清除条件启动。为了避免在中断时返回到主程序中，可以使用 ZS 命令进行程序堆栈，从而使跳转子程序只发生 1 次跳转。

参数：RI n

其中：n=0 清除中断条件启动

 n=1 恢复条件启动

若不带参数，则清除中断条件启动。

用途：

运动进行中	No
在程序中	Yes
命令行	No
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	-

相关命令：

#ININT	中断处理子程序
“II（二进制 EC）”	中断输入

举例：

#A; II1; JP#A; EN	程序标号
#ININT	开始中断子程序
MG “INPUT INTERRUPT”	显示信息
SB1	设置输出行 1
RI1	返回主程序且恢复条件启动

提示：应用程序必须执行#ININT 子程序命令。

131) RL（二进制 DD）

功能：返回最新的锁存位置

说明：RL 命令将返回最新的锁存位置。AL 命令必须先使能锁存，然后 0 出现在相应的输入点。

每个控制轴使用通用输入作为锁存输入，具体如下：

X（A）轴锁存	输入 1
Y（B）轴锁存	输入 2
Z（C）轴锁存	输入 3
W（D）轴锁存	输入 4
E 轴锁存	输入 9
F 轴锁存	输入 10
G 轴锁存	输入 11
H 轴锁存	输入 12

CE 命令可以对锁存使能进行配置。

注：锁存功能需带主编码器工作。当用于步进电机而未带编码器时，锁存用于捕获步进位置。为了捕获步进位置，用户可以从控制器步进（PWM）输出到主编码器输入（channel A+）加根线，从而连接方向（信号）输出到 channel B+输入。使用 CE 命令可以配置步进/方向情况下的主编码器。锁存捕获步进位置是以控制器产生的脉冲为基础。

参数：RL nnnnnnnnnn

其中：n 是 X、Y、Z、W、A、B、C、D、E、F、G、H 或者所要指定轴的任意组合。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	位置格式

操作应用：

_RLn 读取指定轴的锁存位置。



“AL” 锁存使能

JG, 5000	设置 B 轴 JOG 速度为 5000 计数单位/秒
BGB	开始运动
ALB	B 轴锁存使能, 确保大概 2 秒后, 输入变低
RLB	返回最新的锁存位置
10000	

运动进行中	Yes	缺省值	0
在程序中	Yes	缺省格式	位置格式
命令行	Yes		
适用控制器	所有型号控制器		

_RPn 读取指定轴的命令参考位置。

: PF7	位置格式 7
0: RP	
0000200, -0000010, 0000000, -0000110	返回 A、B、C、D 轴参考位置
RPA	
0000200	返回 A 轴参考位置
RPB	
-0000010	返回 B 轴参考位置
PF-6.0	将格式改成十六进制格式
RP	
\$0000C8, \$FFFFFF6, \$000000, \$FFFF93	用十六进制格式返回 A、B、C、D 轴参考位置
Position= _RPA	分配变量 Position 为 RPA 值

224

说明： RS 命令是将控制器运行的状态重新设置为开机时的状态，且先前保存的控制器状态、参数值及保存的程序都能恢复。

用途：

运动进行中	Yes
在程序中	No
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	-

134) 〈control〉 R 〈control〉 S

功能： Master Reset

说明： 此命令将控制器设成出厂状态，且删除控制器的 EEPROM。

控制器在指定标号 MRST 时通过安装跳线来完成主复位及重新设置控制器的参数（电路或复位键），然后拿掉短路棒。

用途：

运动进行中	Yes
在程序中	No
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	-

注：此命令不支持网络连接，若想做尝试的话都将挂断主机。

135) 〈control〉 R 〈control〉 V

功能： 修改信息

说明： 此命令导致控制器返回控件软件修改信息。

用途：

运动进行中	Yes
在程序中	No
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	-

136) SA

功能： 发送命令

说明： SA 命令从主控制器发送命令到从控制器中。任何发送到从控制器的命令都会被从控制器当作本地命令使用。只有一些本地命令必须用 SA 命令发送。

参数： Sah=arg 或 Sah=arg, arg, arg, arg, arg, arg, arg, arg, arg, arg,
其中：h 是用于发送命令到从控制器的端口。

arg 是数、控制器操作、变量、数字或字符串；数字值范围是 4 字节的整数 (2^{31}) 在前，2 字节的小数 (+/-2, 147, 483, 647.9999) 在后。最大的字符串为 6。字符串用引号来识别。

典型的应用是第一个参数作为字符串来使用，例如“KI”，随后的参数作为参数去发送命令，例如：SAF=“KI”，1, 2 就可以发送命令。

用途：

运动进行中	Yes
在程序中	Yes

缺省：

缺省值	-----
缺省格式	-----

命令行	Yes
适用控制器	DMC-2100, DMC-2200

操作应用:

_SAhn 读取 SA 发送命令的响应值。h 值表示端口 A~F, n 值表示从控制器 (1~8) 返回的

指定领域。若未使用指定领域, 操作将是 -2^{31} 。

相关命令:

“MG” 发送信息到端口

举例:

SAA= “KI”, 1, 2 发送命令到端口 A (从控制器): KI1, 2

SAA= “TE” 发送命令到端口 A (从控制器): TE

MG_SAA 显示 _SAA 内容 (对 TE 命令首先响应)

: 132

MG_SAB 显示 _SAA 内容 (对 TE 命令次响应)

: 12

137) SB (二进制 EA)

功能: 设置输出位

说明: SB 命令用于设置输出位。

参数: SB n

其中: n 是整数, 表示指定控制器的位输出被设成高电平 (输出=1)。

注: 当使用 Modbus 装置 (DMC-2100, 2200) 时, 用下列公式来计算 Modbus 装置的 I/O 点:

$$n = (\text{SlaveAddress} * 10000) + (\text{HandleNum} * 1000) + ((\text{Module} - 1) * 4) + (\text{Bitnum} - 1)$$

当 Modbus 装置有从装置与之相连时, 使用从地址, 其地址范围为 0~255。请注意: 对 Modbus 使用从装置非常少见, 因此, 此数值通常为 0。

HandleNum 是端口指定器, 从 A~F。

Module 是模块在机槽中的位置, 从 1~16。

Bitnum 是模块的 I/O 点, 从 1~4。

用途:

运动进行中 Yes

在程序中 Yes

命令行 Yes

适用控制器 所有型号控制器

缺省:

缺省值 -

缺省格式 -

相关命令:

“CB” 清除位

举例:

SB5 输出点 5 设置为高电平

SB1 输出点 1 设置为高电平

138) SC (二进制 E1)

功能：响应电机停止的原因

说明：SC 命令允许用户去查询电机停止的原因，以下是控制器对停止代码做出的响应：

代码	解释	代码	解释
0	电机运行时，独立模式	9	寻边后停止 (FE)
1	在独立位置时，电机停止	10	回零后停止 (HM)
2	通过 FWD 限位开关，减速或停止	11	通过选择急停停止
3	通过 REV 限位开关，减速或停止	50	凸轮运行
4	通过 ST 命令减速或停止	51	凸轮停止
6	通过急停输入停止	99	MC 时间溢出
7	通过急停命令 (AB) 停止	100	电机运行时，矢量程序
8	通过位置错误关断电机 (OE1) 来减速或停止	101	命令矢量时，电机停止

参数：SC nnnnnnnnnn

其中：n 是 A、B、C、D、E、F、G、H 或者所要指定轴的任意组合。

用途：

运动进行中 Yes
 在程序中 Yes
 命令行 Yes
 适用控制器 所有型号控制器

缺省：

缺省值 -
 缺省格式 3.0

操作应用：

_SCn 读取指定轴停止代码的值。

举例：

Tom= _SCD 分配变量 Tom 为 D 停止代码

139) SH(二进制 AA)

功能：激活伺服状态

说明：使用当前电机位置作为命令位置时，SH 命令可以激活伺服状态。

通过电机关断 (MO) 命令来手工调整电机的位置时，可以使用此命令。

参数：SH nnnnnnnnnn

其中：n 是 A、B、C、D、E、F、G、H 或者所要指定轴的任意组合。

用途：

运动进行中 No
 在程序中 Yes
 命令行 Yes
 适用控制器 所有型号控制器

缺省：

缺省值 -
 缺省格式 -

相关命令：

“MO” 电机关断

举例：

SH 激活 A、B、C、D 电机的伺服状态
 SHA 激活 A 电机的伺服状态，B、C、D 保持原来的状态
 SHB 激活 B 电机的伺服状态，A、C、D 保持原来的状态
 SHC 激活 C 电机的伺服状态，A、B、D 保持原来的状态

SHD 激活 D 电机的伺服状态, A、B、C 保持原来的状态

注: SH 命令会改变坐标系。因此, 在使用 SH 命令前的所有位置命令必须重新发命令, 否则控制器会产生错误运动。

140) SP(二进制 92)

功能: 速度

说明: 在独立运动时, 此命令设置任意一些轴或所有轴的转速。

注: 负值将作为绝对值来解释。

参数: SP n, n, n, n, n, n, n, n, n or SPA=n

其中: n 是用于伺服电机的无符号偶数, 范围为 0~12, 000, 000。此值是用编码器计数/秒作为单位。

或

n 是用于步进电机的无符号数, 范围为 0~3, 000, 000。

n=? 返回指定轴的速度。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	25000
缺省格式	位置格式

操作应用:

_SPn 读到指定轴的速度。

相关命令:

"AC"	加速度
"DC"	减速度
"PA"	绝对位置
"PR"	相对位置
"BG"	开始

举例:

PR2000, 3000, 4000, 5000	定义 A、B、C、D 参数
SP5000, 6000, 7000, 8000	定义 A、B、C、D 速度
BG	所有轴启动程序
AM C	完成 C 轴运动后

注: 若为矢量运动时, 可用矢量速度命令 (VS) 去改变速度。SP 不像 JOG (JG) 是一种运动模式, 它不是运动模式。

141) ST(二进制 A1)

功能: 停止

说明: ST 命令用于停止指定轴的运动。电机将减速到停止。

参数: STnnnnnnnnnn

其中: n 是 A、B、C、D、E、F、G、H、N、S、T 或者所要指定轴的任意组合或 I/O 状况。

如果所指定的轴或程序是定义的, 那么将不会停止所执行的程序。

若不带参数, 停止所有轴的运动及正执行的程序。

用途:

缺省:

运动进行中	Yes	缺省值	-
在程序中	Yes	缺省格式	-
命令行	Yes		
适用控制器	所有型号控制器		

相关命令:

“BG”	启动程序
“AB”	急停
“DC”	减速率

举例:

ST A	停止 A 轴运动
ST S	停止联动程序
ST ABCD	停止 A、B、C、D 轴运动
ST	停止程序及 A、B、C、D 轴运动
ST SCD	停止联动 AB 程序及 C、D 轴运动

提示: 使用 AM 命令 (运动完成后) 去等待运动停止。

142) TB

功能: 响应位状态

说明: TB 命令从控制器中返回状态信息作为十进制数。当位设成高电平时, 其字节的每个位分别表示如下情况:

位	情况
位 7	执行应用程序
位 6 (DMC-1200, 1600, 1700)	DMA 有效
位 6 (只有 DMC-2000)	用菊花链写地址
位 5	电子凸轮
位 4	执行误差或限位开关子程序
位 3	输入中断使能
位 2	执行输入中断子程序
位 1 (DMC-1200, 1600, 1700)	第二 FIFO 激活
位 1 (只有 DMC-2000)	N/A
位 0	回声开

参数: TB? 返回字节状态

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	-
缺省格式	1.0

操作应用:

_TBn 读取字节状态。

举例:

65	执行程序并打开回声 ($2^6+2^0=64+1=65$)
----	---------------------------------

143) TC

功能：响应错误码

说明：TC 命令返回 1~255 之间的数，而这个数是个代码，反映了控制器没有接受命令的原因。
 当控制器在命令时停止执行程序或响应给命令 1 个问号时可以使用此命令。TC 命令提供给用户诊断工具。读完 TC 后，错误代码被设成 0。

参数：TCn

其中：n=0 只返回代码
 n=1 返回代码及信息
 n=? 返回错误码

若不带参数，将提供所有轴的错误码。

代码	解释	代码	解释
1	未确认的命令	59	没有匹配的圆括号插入
2	只有命令有效	60	下载错误——行太长或太多行
3	命令在程序中无效	61	复写或错误标号
4	操作错误	62	太多标号
5	输入缓存区满	63	在 IF 语句中无 ENDIF
6	超出范围数	65	IN 命令需有逗号
7	运行时命令无效	66	数组空间满
8	停止时命令无效	67	太多数组或变量
9	变量错误	68	USB 口无效
10	空的程序行或未定义的标号	71	IN 只有在任务#0 时有效
11	无效标号或行数	80	记录模式已运行
12	子程序超过 16 嵌套	81	未指定数组或根源
13	在 JOG 模式下，JG 有效	82	未定义的数组
14	EEPROM 检查和错误	83	无效数
15	EEPROM 写错误	84	太多元素
16	位置移动或减速时出现 IP 错误信号	90	只有 ABCD 有效操作
17	运行程序时，EB、BN、DL 无效	96	步进电机操作时，需安装 SM 跳线
18	电子凸轮时，命令无效	97	坏的二进制命令格式
19	程序串已执行	98	在应用程序中，二进制命令无效
20	电机关断启动无效	99	坏的二进制数
21	运行时启动无效	100	运行 ECAM 时无效
22	不可能由于极限开关启动	101	在 ET 中不合适的指针（必须是 0-256）
24	由于未定义顺序，启动无效	102	在 ECAM 中未定义主轴
25	在 IN 命令时，没有给变量	103	主轴模块大于 256*EP 值
28	S 操作无效	104	轴完成 ECAM 时无效
29	联动时无效	105	先给 EB1 命令
30	程序段太短	110	检测到无霍尔传感器
31	总的移动距离大于 20 亿	111	用 BA 命令做无刷
32	在 1 个程序中，超出 511 个段	112	BZ 命令时间溢出
33	VP 或 CR 不能与 LI 命令混合使用	113	在 BA 命令中未运动

41	电子凸轮记录范围错误	114	BZ 命令远离
42	发送电子凸轮数据太慢	118	控制器有 GL1600 无 GL1800 芯片
46	主、从轴带齿轮	120	Ethernet 传送错
50	没有足够领域	121	错的数据流
51	无效问号	122	Ethernet 输入缓冲区超运行
52	丢失”或字符串太长	123	TCP 失去同步
53	在 {} 中有错误	124	Ethernet 端口已使用
54	字符串中带问号	125	从 IP 地址中无 ARP 响应
55	丢失[或[]	126	关闭 Ethernet 端口
56	数组索引无效或超出范围	127	错误的 Modbus 功能代码
57	不好的功能或数组	128	无效的 IP 地址
58	不好的命令响应（例如_GNX）		

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	3.0

操作应用：

_TC 读取错误代码。

举例：

: GF32	错误命令
? TC	响应错误码
01	未认可的命令

144) TD (二进制 DB)

功能： 响应辅编码器位置

说明： TC 命令返回辅编码器的当前位置。当用步进轴或使用输出比较轴时，使用辅编码器无效。

当使用步进电机操作时，TD 命令返回控制器输出的计数。

参数： TDnnnnnnnnnnnn

其中：n 是 A、B、C、D、E、F、G、H 或者所要指定轴的任意组合。

若不带参数，将提供所有轴的辅编码器的位置。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	位置格式

操作应用：

_TDX 读取辅编码器寄存器值。

相关命令：

“DE” 辅编码器

举例：

```

: PF7           位置格式 7
: TD           返回 A、B、C、D 轴辅编码器
0000200, -0000010, 0000000, -0000110
TDA           返回 A 轴辅编码器
0000200

DUAL= _TDA     分配变量 DUAL 为 TDA 值

```

145) TE (二进制 DA)

功能：响应误差

说明：此命令返回电机的当前位置误差，其误差范围是 2147483647。此命令对于步进电机操作无效，因为步进电机采用开环方式。

参数：TEnnnnnnnnnn

其中：n 是 A、B、C、D、E、F、G、H 或者所要指定轴的任意组合。
若不带参数，将提供所有轴的位置误差。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	0
缺省格式	位置格式

相关命令：

“OE”	位置错误关断电机
“ER”	最大误差极限
#POSERR	错误处理子程序
“PF”	位置格式

举例：

```

TE           返回所有轴的位置误差
00005, -00002, 00000, 00006
TEA         返回 A 轴的位置误差
00005

TEB         返回 B 轴的位置误差
-00002

Error= _TEA  分配变量 Error 为 A 轴位置误差

```

提示：通常在伺服控制下，位置误差很小。在加速度时，位置误差是最大的。

146) TH

功能：响应端口状态

说明：TH 命令用于询问控制器的端口状态。从此命令返回的数据中可以指出当前控制器 IP 地址及 Ethernet 地址。此数据跟在每个端口状态之后指出连接类型、IP 地址及是否是 QW 或用 HC 命令配置的命令端口。

参数：无

用途：

缺省：

运动进行中	Yes	缺省值	-----
在程序中	Yes	缺省格式	-----
命令行	Yes		
适用控制器	DMC-2100, DMC-2200		

相关命令:

“IH”	Internet 端口
“HR”	端口恢复
“WH”	指定端口

举例:

: TH 响应当前端口配置

```
CONTROLLER IP ADDRESS 10, 51, 0, 87 ETHERNET ADDRESS 00-50-4C-08-01-1F
IHA TCP PORT 1050 TO IP ADDRESS 10, 51, 0, 89 PORT 1000 SLAVE CD COMMAND
IHB TCP PORT 1061 TO IP ADDRESS 10, 51, 0, 89 PORT 1001 SLAVE CD AQ
IHC TCP PORT 1012 TO IP ADDRESS 10, 51, 0, 93 PORT 1002 SLAVE EF COMMAND
IHD TCP PORT 1023 TO IP ADDRESS 10, 51, 0, 93 PORT 1003 SLAVE EF QW
IHE TCP PORT 1034 TO IP ADDRESS 10, 51, 0, 101 PORT 1004 SLAVE IOC COMMAND
IHF TCP PORT 1045 TO IP ADDRESS 10, 51, 0, 101 PORT 1005 SLAVE IOC QW
IHG AVAILABLE
IHH AVAILABLE
```

147) TI (二进制 EO)

功能: 响应输入点

说明: 这个命令返回输入状态, 包括扩展 I/O 的状态。此命令返回的值是个十进制数, 用 8 位值表示 (十进制范围从 0~255)。每个位代表 1 个输入, 其中 LSB 是最低电平输入数, MSB 是最高电平输入数。

参数: TIn

其中: n=0	返回输入 1~8 的输入状态
n=1	返回输入 9~16 的输入状态 (见注 1)
n=2	返回输入 17~24 的输入状态 (见注 2)
n=3~9	n 表示扩展输入范围是 (8*n) +1~ (8* (n+1)) (见注 3)
n=10	返回输入 81~88 (辅编码器输入) 的输入状态 (见注 4)
n=11	返回输入 89~96 (辅编码器输入) 的输入状态 (见注 4)

若不带参数, 将返回输入 1~8 的输入状态

注 1: 只有控制器多于 4 轴时才应用

注 2: 不适用于 DMC-2x00 系列控制器

注 3: 当使用扩展 I/O 作为输入时才使用

注 4: 辅编码器输入变量时才使用

用途:

运动进行中	Yes	缺省值	-
在程序中	Yes	缺省格式	1.0
命令行	Yes		
适用控制器	所有型号控制器		

缺省:

操作应用:

_Tin 读取 ‘n’ 指定的输入块字节的状态。注意此操作只有指定位信息才能屏蔽返回。

举例：

TI
09 输入 4 为高电平，其它为低电平
TI
00 所有的输入点为低电平
Input= _TI 设置变量输入为 TI 值
TI
255 所有的输入点为高电平

148) TIME*

功能：TIME 操作

说明：TIME 操作返回运动的实时值。此值建立在 TM 命令基础上，表示伺服循环更新数。TM 命令缺省值为 1000。随着更新率的增加，TIME 操作将以 1 计数/大约 1000usec 更新率增加。注意：更新率 1000 值将是实际设置 1/1024 秒的更新率。因此 TIME 操作返回的值将以实际时间的 2.4% 削减。
在标准复位或主复位时，时钟被设成 0。

TIME 在操作时与其它命令的操作有所不同，不需要带 “_”。

举例：

MGTIME 显示内部时钟值

149) TL

功能：转矩极限

说明：TL 命令用于设置电机输出时的极限。例如，TL5，说明了电机输出极限为 5V，电机的最大输出为 9.998V。

参数：TL n, n, n, n, n, n, n, n or TLA=n

其中：n 是 0~9.998 之间的无符号数，分辨率为 0.003V。

n=? 返回指定轴的转矩极限值。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	1.0

操作应用：

_TLn 读取指定轴的转矩极限值。

举例：

TL1, 5, 9, 7.5	A、B、C、D 轴的输出极限分别为 1V, 5V, 9V, 7.5V
TL?, ?, ?, ?	返回所有轴的极限值
1.0000, 5.0000, 9.0000, 7.5000	
TL?	返回 A 轴极限值
1.0000	

150) TM (二进制 E5)

功能： 取样时间

说明： TM 命令设置了控制循环的取样周期。若改变这个取样周期，将无法校准速度及加速度参数。若为负数，将关闭内部时钟，但可以用外部设备作为时间基础来使用。此命令的单位是 μsec 。

参数： TMn

其中：**带快速控件：** n 是 125~20000 之间的十进制整数，分辨率为 125 微秒。当使用快速控件时，DMC-1210 及 DMC-1710 的采样时间最小。在快速控件模式下，没有齿轮、CAM、PL、模拟前馈、步进、主域的条件启动、DMA、TV 等功能。使用快速控件，最小的取样时间如下所示：

1~2 轴 Optima 系列控制器	125 μsec
3~4 轴 Optima 系列控制器	250 μsec
5~6 轴 Optima 系列控制器	375 μsec
7~8 轴 Optima 系列控制器	500 μsec

带普通控件： 使用普通控件，最小的取样时间如下所示：

1~2 轴 Optima 系列控制器	250 μsec
3~4 轴 Optima 系列控制器	375 μsec
5~6 轴 Optima 系列控制器	500 μsec
7~8 轴 Optima 系列控制器	625 μsec

n=? 返回取样时间值。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	1000
缺省格式	1.0

操作应用：

_TM 读取指定轴的取样时间值。

举例：

TM-1000	关闭内部时钟
TM2000	设置采样率到 2000[EQN “[μ]”]sec（这将使速度按二分之一切断，加速度按四分之一切断）
TM1000	返回缺省采样率

151) TN (二进制 B4)

功能： 正切

说明： TNm, n 描述正切轴的运动轨迹。m 是正切轴的缩放系数，用计数/度表示；n 是正切轴的绝对位置，其中正切轴在坐标系中以 0 度作为原点。正切轴用命令 VMn, m, p 来指定，其中 p 是正切轴。在切割刀具必须与工件保持正切关系的切割应用中，使用此函数非常有用。

参数： TN m, n

其中：m 是缩放系数，单位是计数/度，范围为 -127~127 之间，小数分辨率为 0.004。

m=? 返回正切轴的第一位置值。

当用步进电机操作时，m 是缩放系数，单位是步/度。

n 是正切角为 0 时的绝对位置，范围是 $\pm 2 \times 10^9$ 。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	-
缺省格式	--

操作应用:

_TN 读取正切轴的第一位置值。这要求用户在运动开始前对正切轴进行正确定位。

相关命令:

"VM"	联动运动模式
"CR"	圆弧命令

举例:

VMA, B, C	指定 A、B 轴为联动运动方式, C 轴为正切轴
TN100, 50	当正切角为 0 时, 指定 A、B 轴的缩放系数分别是 100 计数/度、50 计数/度
VP1000, 2000	指定 A、B 轴的矢量位置
VE	结束矢量运动
BGS	正切轴开始运动

152) TP (二进制 D9)

功能: 响应位置

说明: 此命令用于返回电机的当前位置。

参数: TPnnnnnnnnnn

其中: n 是 A、B、C、D、E、F、G、H 轴或任意结合。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	-
缺省格式	--

操作应用:

_TPx 读取指定轴的当前位置值。

相关命令:

"PF"	位置格式
------	------

举例: 定义 A 轴位置为 200 (十进制), B 轴位置为 -10 (十进制), C 轴位置为 0, D 轴位置为 -110 (十进制)。此值返回的参数是 4 倍频的计数单位。

: PF7	位置格式 7
: TP	返回 A、B、C、D 轴位置
0000200, -0000010, 0000000, -0000110	
TPA	返回 A 轴电机位置
0000200	
TPB	返回 B 轴电机位置
-0000010	
PF-6.0	改成十六进制格式

TP 用十六进制数返回 A、B、C、D 轴位置
 \$0000C8, \$FFFFFF6, \$000000, \$FFFF93
 Position= _TPA 分配变量 Position 为 TPA 值

153) TR

功能：响应轨迹

说明：TR 命令使程序中的每条指示在执行前先发送给通讯口。TR1 有这种功能，TR0 没有。此命令对于移去程序中的错误非常有用。

参数：TRn

其中：n=0 没有响应轨迹功能
 n=1 能够响应轨迹功能
 若不带参数，就没有响应轨迹功能

用途：

运动进行中	Yes	缺省：	缺省值	TR0
在程序中	Yes		缺省格式	--
命令行	Yes			
适用控制器	除 DMC-2100, DMC-2200 (ETHERNET) 外的所有型号控制器			

154) TS (二进制 DF)

功能：响应开关

说明：TS 命令返回回零开关、前馈限位开关、反向限位开关、误差条件、运动条件及电机状态等信息给控制器。返回值是个十进制数，用 8 位值表示（十进制范围：0~255）。其每个位代表的含义如下表所示：

位	解释
位 7	若高电平，轴运动
位 6	若高电平，轴误差超出误差极限
位 5	若高电平，关断电机
位 4	未定义
位 3	若高电平，前馈限位开关无效
位 2	若高电平，反向限位开关无效
位 1	回零 A 开关
位 0	锁存

注：对于有效高电平或有效低电平配置，当开关无效时，位为 1；若开关有效时，位为 0。

参数：TSnnnnnnnnnn

其中：n 是 A、B、C、D、E、F、G、H 或者所要指定轴的任意组合。
 若不带参数，提供所有轴的状态。

用途：

运动进行中	Yes	缺省：	缺省值	-
在程序中	Yes		缺省格式	3.0
命令行	Yes			
适用控制器	所有型号控制器			

操作应用：

_TS 读取开关的当前状态。

举例：

V1=	_TSB	分配 TSB 值给变量 V1
V1=		查询变量 V1 值
015 (返回值)		与位模式相对应的十进制值 00001111
		Y 轴未运动 (位 7---有 0 值)
		Y 轴未超出误差极限 (位 6 有 0 值)
		Y 轴运动开 (位 5 有 0 值)
		Y 轴前馈限位无效 (位 3 有 1 值)
		Y 轴反向限位无效 (位 2 有 2 值)
		Y 轴回零开关为高电平 (位 1 有 1 值)
		Y 轴 (位 0 有 1 值)

155) TT (二进制 DE)

功能：响应转矩

说明：TT 命令指出模拟输出信号值，且这个值为-9.998~9.998V 之间。

参数：TTnnnnnnnnnn

其中：n 是 A、B、C、D、E、F、G、H 或者所要指定轴的任意组合。

若不带参数，提供所有轴的转矩。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	-
缺省格式	1.4

操作应用：

_TTn 读取指定轴的转矩值。

相关命令：

"TL" 转矩极限

举例：

V1=	_TTA	分配 TTA 值给变量 V1
TTA		A 轴转矩为-.2843V
-0.2843		

156) TV (二进制 DC)

功能：响应速度

说明：TV 命令返回轴的实际速度，此值是以编码器的计数/秒作为单位。返回值含符号。

参数：TVnnnnnnnnnn

其中：n 是 A、B、C、D、E、F、G、H 或者所要指定轴的任意组合。

若不带参数，提供所有轴的辅编码器位置。

用途：

缺省：

运动进行中	Yes	缺省值	-
在程序中	Yes	缺省格式	7.0
命令行	Yes		
适用控制器	所有型号控制器		

操作应用：

_TVn 读取指定轴的速度值。

举例：

VELA= _TVA 分配 A 轴速度值给变量 VELA

TVA 返回 A 轴速度

0003420

注：TV 命令使用特殊的平均滤波来计算（最大值.25sec）。因此，TV 返回的是平均速度而不是瞬时速度。

157) TW (二进制 CA)

功能：时间溢出

说明：如果 MC 命令有效及电机在完成运动包路线后 n msec 内不在或超出实际位置时，TW 命令会设置时间溢出且发出错误提示。若出现时间溢出，那么将会清除 MC 条件启动，且停止代码被设成 99。一个应用程序将跳到特殊的标号#MCTIME。RE 命令用于从#MCTIME 子程序返回。

参数：TW n, n, n, n, n, n, n, n or TWA=n

其中：n 指定在 mces 内的时间溢出，其范围是 0c32767msec 之间。

n=-1 没有时间溢出

n=? 返回指定轴在 msec 内的时间溢出值。

用途：

运动进行中	Yes	缺省：	
在程序中	Yes	缺省值	32766
命令行	Yes	缺省格式	
适用控制器	所有型号控制器		

操作应用：

_TWn 读取指定轴在 msec 内的时间溢出值。

相关命令：

“MC” 定位完毕

158) TZ

功能：响应 I/O 状态

说明：TZ 命令用于查询 I/O 状态。

参数：TZ

用途：

运动进行中	Yes	缺省：	
在程序中	Yes	缺省值	-----
命令行	Yes	缺省格式	-----

适用控制器 DMC-2100, DMC-2200

相关命令:

TI	响应输入点
SB/CB	设置/清除输入位
OP	输出通信口
CO	配置 I/O

举例:

: TZ 响应当前主 I/O 状态

BLOCK 0 (8-1) 用作输入一值 255 (1111 - 1111)

BLOCK 0 (8-1) 用作输出一值 0 (0000 - 0000)

BLOCK 1 (16-9) 用作输入一值 255 (1111 - 1111)

BLOCK 1 (16-9) 用作输出一值 0 (0000 - 0000)

BLOCK 2 (24-17) 配置为输入一值 255 (1111 - 1111)

BLOCK 3 (32-25) 配置为输入一值 255 (1111 - 1111)

BLOCK 4 (40-33) 配置为输入一值 255 (1111 - 1111)

BLOCK 5 (48-41) 配置为输入一值 255 (1111 - 1111)

BLOCK 6 (56-49) 配置为输入一值 255 (1111 - 1111)

BLOCK 7 (64-57) 配置为输入一值 255 (1111 - 1111)

BLOCK 8 (72-65) 配置为输入一值 255 (1111 - 1111)

BLOCK 9 (88-81) 用作输入一值 255 (1111 - 1111)

BLOCK 10 (96-89) 用作输入一值 255 (1111 - 1111)

159) UI**功能:** 用户中断**说明:** UI 命令引发所选 IRQ 行中断。有 16 个用户中断为 Uin, n=0~15。

使用 UI 命令前, 必须使控制器中一个 IRQ 行使能, 且数据 2 和 4 写入地址 N+1 的控制寄存器。通过 Plug 和 Play 特性可以使中断使能。在您的主程序中, 也必须配备中断服务程序。通过将 6 写到 N+1, 然后读地址 N+1 来读取中断条件。详细说明请参考《用户手册》。

参数: Uin

其中: n 是 0~15 之间的整数。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	除 DMC-2000, 2100, 2200 之外的所有型号控制器

缺省:

缺省值	0
缺省格式	-

举例:

#I	标号
PR 10000	相对位置
SP 5000	速度
BGA	开始运动
AS	等待速度到达
UI1	发送中断 1
EN	程序结束

此程序发送中断到选择的 IRQ 行。主程序写 6 到地址 N+1 中，然后读地址 N+1 来接收对应于 UI1 的数据 E1。

160) UL

功能: 上载

说明: UL 命令将数据通过串口 1 从控制器传送到主计算机。不用行号传送程序。
上载的程序用<Control>Z 或 \ 进行跟踪至文本标志的结尾。

参数: 无

用途:

运动进行中	Yes
在程序中	No
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	0
缺省格式	-

操作应用:

当用作运算时，_UL 给出可用变量数。可用变量数为 254。

相关命令:

"DL"	下载
------	----

举例:

UL;	开始上载
#A	0 行
NO	1 行
NO	2 行
EN	3 行
<Cntrl>Z	终止器

161) VA (二进制 B7)

功能: 矢量加速度

说明: 此命令设置联动运动程序中的矢量加速度。

参数: VA s, t

其中: s 和 t 是 1024~68431360 之间的无符号整数。s 代表 S 坐标系中的矢量加速度，t 代表 T 坐标系中的矢量加速度。参数输入将以 1024 为系数向下取整。参数的单位为

计数单位/ sec^2 。

s=? 查询 S 坐标系矢量加速度值。

t=? 查询 T 坐标系矢量加速度值。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	256000
缺省格式	位置格式

操作应用:

_VAx 读取所指定轴的矢量加速度值。

相关命令:

“VS (二进制 B9)”	矢量速度
“VP (二进制 B2)”	矢量位置
“VE”	矢量结束
“CR (二进制 B3)”	圆弧
“VM”	矢量方式
“BG (二进制 A0)”	开始程序
“VD (二进制 B8)”	矢量减速度
“VT (二进制 B6)”	矢量平滑常数—S 曲线

举例:

VA 1024	设置矢量加速度为 1024 计数单位/ sec^2
VA ?	查询矢量加速度
00001024	
VA20000	设置矢量加速度
VA ?	查询矢量加速度
0019456	
ACCEL=_VA	分配变量 ACCEL 为 VA 值

162) VD (二进制 B8)

功能: 矢量减速度

说明: 此命令设置联动运动程序中的矢量减速度。

参数: VD s, t

其中: s 和 t 是 1024~68431360 之间的无符号整数。s 代表 S 坐标系矢量减速度, t 代表 T 坐标系矢量减速度。参考输入将以 1024 为系数向下取整。参数单位为计数单位/ sec^2 。

s=? 查询 S 坐标系矢量加速度

t=? 查询 T 坐标系矢量加速度。

用途:

运动进行中	No
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	256000
缺省格式	位置格式

操作应用:

_VDn 读取所指定坐标系 S 或 T 矢量加速度值。

相关命令：

“VA（二进制 B7）”	矢量加速度
“VS（二进制 B9）”	矢量速度
“VP（二进制 B2）”	矢量位置
“VE”	矢量结束
“CR（二进制 B3）”	圆弧
“VM”	矢量方式
“BG（二进制 A0）”	开始程序
“VT（二进制 B6）”	矢量平滑常数—S 曲线

举例：

#VECTOR	矢量程序标号
VM AB	指定运动平面
VA 1000000	矢量加速度
VD 5000000	矢量减速度
VS 2000	矢量速度
VP 10000, 20000	矢量位置
VE	矢量结束
BGS	开始运动

163) VE

功能： 矢量程序结束符

说明： 需要用 VE 来指定一个联动运动程序的结束。在程序中，VE 跟在 VP 或 CR 命令的后面。
VE 等效于 LE 命令。

VE 命令适用于有坐标系选择 S 和 T 的矢量运动。要想选择坐标系，请使用 CAS 或 CAT 命令。

参数： VEn

无参数表明矢量程序结束。

n=? 查询矢量长度，单位为计数单位。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	—
缺省格式	—

操作应用：

_VEn 读取所指定坐标系 S 和 T 矢量长度，单位为计数单位。

相关命令：

“VA（二进制 B7）”	矢量加速度
“VS（二进制 B9）”	矢量速度
“VP（二进制 B2）”	矢量位置
“CR（二进制 B3）”	圆弧
“VM”	矢量方式
“BG（二进制 A0）”	开始程序
“VD（二进制 B8）”	矢量减速度
“VT（二进制 B6）”	矢量平滑常数—S 曲线

“CS” 清除程序

举例:

VM AB	AB 平面矢量运动
VP 1000, 2000	直线插补线段
CR 0, 90, 180	圆弧插补线段
VP 0, 0	直线插补线段
VE	结束矢量程序
BGS	开始运动

164) VF

功能: 变量格式

说明: VF 命令查询控制器要显示的数字位格式进行定义。

如果数字超过此格式, 该数就显式为最大可能的正数或负数 (即 999.99, -999, \$8000 或 \$7FF)。

参数: VF m, n

其中: m 和 n 是无符号数, $0 < m < 10$, $0 < n < 4$

m 表示小数点之前的位数, 负的 m 指定为十六进制格式。当为十六进制时, 字符串用 \$ 符号引导, 并以 2 的补码进行显示, 第 1 位为符号位。

n 表示小数点之后的位数。

m=? 查询变量和数组的格式值。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	10.4
缺省格式	2.1

操作应用:

_VF 读取变量和数组的格式值。

相关命令:

“PF” 位置格式

举例:

VF 5.3	设置 5 位整数位, 3 位小数位
VF 8.0	设置 8 位整数位, 无小数位
VF -4.0	指定为十六进制, 小数点左侧为 4 个字节

165) VM

功能: 联动运动方式

说明: VM 命令指定联动运动方式和运动平面。对于两轴任意组合的运动, 可以用此命令来指定。

用指令 VP 和 CR 指定直线插补和圆弧插补运动。在 BGS 或 BGT 之前可以写入多达 511 个程序段。在运动期间, 当缓冲区空出空间时, 可以给定附加程序段。用户必须在缓冲区留有足够程序段以确保连续运动。在最后一个程序段之后, 必须写入矢量结束 (VE) 命令, 使控制器适当减速。VM 命令适用于 S 坐标系或 T 坐标系。为了选择坐标系, 请使用 CAS 或 CAT 命令。

参数: VM m, n, p

其中：n 和 m 指定矢量运动平面，可以是任意两个轴。通过将第二参数 m 指定为 N，就能够对一个轴指定矢量运动。指定一个轴的矢量运动对于得到 1 个轴的正弦运动特别有用。

p 是正切轴，可以对任意一个轴进行指定，对于参数 p 来说，N 的值使正切功能关断。

n=? 查询能够发送到缓冲区的运动程序段的可用空间。

0 意味着缓冲区已满，不可以发送附加程序段。

用途：

运动进行中	No
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	A, B
缺省格式	-

操作应用：

_VMn 读取所指定的坐标系 S 或 T 瞬时命令的矢量。

相关命令：

“VA (二进制 B7)”	矢量加速度
“VS (二进制 B9)”	矢量速度
“VP (二进制 B2)”	矢量位置
“VD (二进制 B8)”	矢量减速度
“CR (二进制 B3)”	圆弧
“VE”	矢量结束
“CS”	清除程序
“VT (二进制 B6)”	矢量平滑常数—S 曲线
“AV”	用于矢量距离的条件启动

举例：

CAS	指定 S 坐标系
VM A, B	指定 A, B 轴为联动运动方式
CR 500, 0, 180	指定圆弧插补程序段
VP 100, 200	指定直线插补程序段
VE	结束矢量
BGS	开始运动程序

166) VP (二进制 B2)

功能： 矢量位置

说明： VP 命令定义两轴运动程序中的直线插补程序段的目标坐标值，即两轴插补运动由 VM 命令来选择。此值的单位为 4 倍频计数单位，它是用命令 VS 设定的矢量比例因子的函数。对于 3 轴以上直线插补运动，请使用 LI 命令。

VP 命令适用于所选择的坐标系 S 或 T。要想选择坐标系，请使用命令 CAS 或 CAT。

参数： VP n, m<o>p

其中：n 和 m 是-2147483648~2147483647 之间的带符号整数。每个程序段的长度必须限定在 8×10^6 之内。n 和 m 值将指定坐标系矢量运动值。

o 指定在执行矢量程序段的矢量速度。对于伺服电机运行来说，o 是 0~12000000 之间的无符号偶整数；对步进电机而言，o 是 0~3000000 之间的无符号偶整数。

p 指定在结束矢量程序段运动时所要达到的矢量速度，p 是 0~8000000 之间的无符号偶整数。

用途:

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省:

缺省值	-
缺省格式	-

操作应用:

_VPn 读取程序中上一条指令处的控制轴的绝对坐标值。例如，在第一个运动程序段期间，此命令读取程序开始处的坐标值。用作操作数在直线插补方式 LM 以及在矢量插补方式 VM 中是有效的。

相关命令:

“CR (二进制 B3)”	圆弧
“VM”	矢量方式
“VA (二进制 B7)”	矢量加速度
“VD (二进制 B8)”	矢量减速度
“VE”	矢量结束
“VS (二进制 B9)”	矢量速度
“BG (二进制 A0)”	开始程序
“VT (二进制 B6)”	矢量平滑常数—S 曲线

举例:

#A	程序 A
VM AB	指定运动平面
VP 1000, 2000	指定 A, B 轴矢量位置
CR 1000, 0, 360	指定圆弧插补运动
VE	矢量结束
VS 2000	指定矢量速度
VA400000	指定矢量加速度
BGS	开始矢量运动
EN	程序结束

提示: 自动运动程序中的第一个矢量定义该运动程序的原点。关于运动程序的起点程序中的所有其它矢量均由其终点进行定义。与程序无关的轴不需用闭逗号去分离。

167) VR (二进制 BA)

功能: 矢量速度比

说明: VR 设置当前矢量速度的倍率。矢量速度由命令 VS 或与 CR、VP 和 LI 命令一起使用的算式 ‘<’ 和 ‘>’ 来设置。VR 立即起作用且会使所有后续矢量速度命令产生比例。VR 不对加速度或减速度起比例作用，但速度的改变由 VA 和 VD 所指定的加速度或减速度来实现。

参数: VR s, t

其中: s 和 t 是 0~10 之间的数，分辨率为 0.0001。由 s 指定的值是适用于 S 坐标系的矢量速度比例，而 t 是适用于 T 坐标系的矢量速度比例值。

s=? 查询 S 坐标系矢量速度比例值。

t=? 查询 T 坐标系矢量速度比例值。

用途:

运动进行中	Yes
-------	-----

缺省:

缺省值	1
-----	---

在程序中	Yes	缺省格式	-
命令行	Yes		
适用控制器	所有型号控制器		

操作应用：

_VRn 读取指定坐标系 S 或 T 的矢量速度比例。

相关命令：

“VS (二进制 B9)” 矢量速度

举例：

#A	矢量程序
VM AB	矢量方式
VP 1000, 2000	矢量位置
CR 1000, 0, 360	指定圆弧插补运动
VE	矢量结束
VS 2000	矢量速度
BGS	开始矢量运动程序
AMS	运动完成后
JP#A	重复运动
#SPEED	速度倍率
VR @AN[1] 0.1	读取模拟输入计算比例
XQ#A, 0; XQ#SPEED, 1	同时执行任务 0 和任务 1

注意：VR 命令用于进位倍率调整，特别适用于用算式 ‘<’ 和 ‘>’ 指定个别程序段速度时。

168) VS (二进制 B9)

功能：矢量速度

说明：VS 命令指定以 LM 或 VM 方式联动运动程序中的矢量速度。对于矢量或直线插补运动来说，矢量速度可以采用各指定轴的速度的平方值之和的平方根来计算。

参数：VS s, t

其中：s, t 是无符号偶数，对于伺服电机来说，其范围为 2~12000000，对于步进电机而言，其范围为 2~3000000。s 是用于 S 坐标系的速度，而 t 是用于 T 坐标系的速度。单位为计数单位/sec。

s=? 查询 S 坐标系矢量速度值。

t=? 查询 T 坐标系矢量速度值。

用途：

运动进行中	Yes	缺省：	缺省值	25000
在程序中	Yes		缺省格式	-
命令行	Yes			
适用控制器	所有型号控制器			

操作应用：

_VSn 读取指定坐标点 S 或 T 中的矢量速度。

相关命令：

“VA (二进制 B7)”	矢量加速度
“VP (二进制 B2)”	矢量位置
“CR (二进制 B3)”	圆弧插补运动
“LM (二进制 B0)”	直线插补运动

“VM”	矢量方式
“BG (二进制 A0)”	开始程序
“VE”	矢量结束

举例：

VS 2000	定义 S 坐标系矢量速度
VS ?	查询 S 坐标系矢量速度
002000	

提示： 可以将矢量速度附加到单个矢量程序段。更为详尽的信息，请参见 VP，CR，LI 命令说明。

169) VT (二进制 B6)

功能： 矢量时间常数—S 曲线

说明： VT 命令对 VM，LM 型矢量运动中的加，减速度函数进行滤波，以产生平滑的速度仓络线。最终的仓络线（称之为平滑），具有连续加速度，并使机械振动减。VT 设置滤波器的带宽，其中 1 表明无滤波，0.004 意味着最大滤波。注意：平滑滤波会导致运动时间延长。

参数： VT s, t

其中：s, t 是无符号数，范围为 0.004~1，分辨率为 1/256。S 值用于 S 坐标系，t 值用于 T 坐标系。

s=? 查询 S 坐标系矢量时间常数值。

t=? 查询 T 坐标系矢量时间常数值。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	1.0
缺省格式	1.4

操作应用：

_VTn 读数指定坐标系中的矢量时间常数

相关命令：

“IT (二进制 93)” 用平滑独立运动的独立时间常数。

举例：

VT 0.8	设置 S 坐标系矢量时间常数
VT ?	查询 S 坐标系矢量时间常数
0.8	

170) WC (二进制 D4)

功能： 等待轮廓数据

说明： WC 命令在轮廓方式中起标志作用。在执行此命令后，直到内部轮廓数据缓冲区准备好接收新命令，控制器才接收新数据。此命令避免了轮廓数据在轮廓数据缓冲区重复写入。

用途：

运动进行中	Yes
在程序中	Yes
命令行	Yes
适用控制器	所有型号控制器

缺省：

缺省值	1.0
缺省格式	1.4

相关命令:

“CM (二进制 BD)”	轮廓方式
“CD (二进制 BE)”	轮廓方式
“DT (二进制 BF)”	轮廓方式

举例:

CM ABCD	指定轮廓方式
DT 4	指定轮廓时间增量
CD 200, 350, -150, 500	指定 A, B, C, D 轴增量位置, A 轴运动 200 计数单位, B 轴运动 350 计数单位, C 轴运动-150 计数单位, D 轴运动 500 计数单位。
WC	等待轮廓数据完成。
CD 100, 200, 300, 400	
WC	等待轮廓数据完成
DT0	停止轮廓
CD 0, 0, 0, 0	退出轮廓方式

171) WH

功能: 指定端口

说明: 用 WH 命令来识别执行命令的端口。命令返回 IHA~IHH 以指示执行命令的所属端口。如果是串口。如果是串口通信, 此命令则返回 RS232。

参数: 无

用途:

运动进行中	Yes	缺省:	缺省值	-----
在程序中	Yes		缺省格式	-----
命令行	Yes			
适用控制器	DMC-2100, DMC-2200			

操作应用:

_WH 读取执行命令的端口的数符。端口 A~HET 0~7 来表示, 而代表串口。

相关命令:

“TH”	报告端口
------	------

举例:

: WH	请求端口识别
IHC	命令在端口识别
: WH	请求端口识别
RS232	命令在 RS32 口执行

172) WT (二进制 D3)

功能: 等待

说明: WT 命令是用于对事件定时的条件启动命令。在执行此命令之后, 控制器在执行下一条命令之前等待所指定的采样周期数。如果没有用 TM 命令更改采样周期 (原为 1msec), 那么等待命令的单位仍为 msec。

参数: WTn

其中: n 是 $0 \sim 2 \times 10^8$ 十进制数。

用途:

缺省:

运动进行中	Yes	缺省值	-
在程序中	Yes	缺省格式	-
命令行	Yes		
适用控制器	所有型号控制器		

举例：

假设在运动之后过 10sec，必须使继电器闭合。

```
#A          程序 A
PR50000     相对位置运动
GBA         开始运动
AMA         在运动完成之后
WT 10000    等待 10sec
SB0         接通继电器
EN          结束程序
```

提示：要想得到更长等待时间，就写入多条 WT 命令。

173) XQ

功能：执行程序

说明：XQ 命令开始执行驻留在控制器程序存储器中的程序。执行将以标号或指定的行号开始。控制器可以用同时执行 8 个程序。

参数：XQ#A, n XQ m, n

其中：A 是程序名，可以有 7 个字符

m 是行号

n 是多任务中的域号，是个 0~7 范围中的整数。

注意：命令 XQ 的参数是可选参数，如果不指定参数，就执行域 0 存储器中的第一个程序。

用途：

缺省：

运动进行中	Yes	缺省值	0
在程序中	Yes	缺省格式	-
命令行	Yes		
适用控制器	所有型号控制器		

操作应用：

_XQn 读取在域 n 中执行的当前行号，若域 n 未运行，则为-1。

相关命令：

“HX” 暂停执行

举例：

```
XQ#APPLE, 0      开始执行域 0 中标号为“APPLE”的程序
XQ#DATA, 2       开始执行域 2 中标号为“DATA”的程序
XQ 0             开始执行 0 行中的程序
```

提示：在执行程序之前，请不要忘记退出编辑方式。

174) ZS

功能：程序堆栈

说明：ZS 命令只在应用程序中有效，用它来避免从中断（输入或报警）返回。ZS 只将堆栈返回到其初始状态。ZS1 调整堆栈消除一次返回。这就将跳转移交给跳转子程序。当使用

ZS 时，不要使用 RI（从中断返回）命令。要重新使中断使能，您必须再次使用 II 命令。

堆栈的状态能够用_ZSn 来查询，请参见操作应用。

参数： ZSn

其中： n=0 使堆栈返回到初始条件。
 n=1 消除堆栈一次返回

用途：

运动进行中 Yes
 在程序中 Yes
 命令行 No
 适用控制器 所有型号控制器

缺省：

缺省值 0
 缺省格式 3.0

操作应用：

_ZSn 读取所指定域的堆栈级别，其中 n=0, 1, 2 或 3。注意：用 A（域 0），B（域 1），C（域 2），D（域 3）也能指定 n。

举例：

II 1	输入口 1 产生输入中断
#A; JP#A; EN	主程序
#ININT	输入中断
MG “INTERRUPT”	发送信息
S= _ZS	查询堆栈
S=	打印堆栈
ZS	0 堆栈
S= _ZS	查询堆栈
S=	打印堆栈
EN	结束

第三章 附录

1. 典型应用领域：

航 天 食 品 加 工 机 床	天线定位控制	空间摄像控制	天文望远镜
	食品包装	家禽修整加工机	精密切肉机
	无心磨床	EDM 机床	螺纹机床
	激光切割机床	铣床	冲压机床
产 业 制 造	快速成型机	超声焊接机	水射流切割
	粘胶配料	软管编织机	子午胎成型机
	绕线机	光纤玻璃拉伸机	同步飞剪
材料输送设备	龙门式输送臂	玻璃净化炉	
	纸板运送	输送机驱动	多轴机械手

医疗器械	核反应棒拆卸搬运	包装系统	码垛机
	人工咀嚼仿真器	血液分析仪	DNA 测试仪
	CAT 扫描仪	测步仪	医疗成像声纳
	尿样测试仪		
半导体测试及加工	晶片自动输送	盒带搬运	IC 插装机
	电路板特型铣	晶片抛光机	晶片探测机
	晶片切割机	IC 引线焊接机	IC 编带机
	SMT 机		
测试及测量	坐标检验	齿轮检验	来料检验
	键盘测试	PCB 测试	焊点超声扫描检查
	显微仪定位控制		
纺织印染机械	自动织袋机	地毯编织机	平网/圆网印花机
	珩缝机	纱锭卷绕机	

2. 相关链接产品:

日本安川 ΣII 、 $\Sigma-V$ 系列全数字交流伺服系统 (30w-55kw)
 宝伦 PDC/PAC 系列交、直流伺服系统 (1kw-7kw)
 德国 WITTENSTEIN 高精度行星齿轮减速器 (1 arcmin-15arcmin)
 韩国 INCOM 一体化集成直线导轨
 日本安川 DD motor (直驱电机)
 美国 PARKER 微型集成 X-Y 工作台、电动缸、直线电机

3. 其它产品:

PNC2000MC 雕铣机数控系统
 PNC2000PB 弯管机数控系统
 PNC2000SW 单针绗缝数控系统
 PNC2000QS 淬火机床数控系统
 PNC2000DS 多组份涂胶/点胶数控系统
 PNC2000GC 玻璃/石材切割数控系统
 PNC2000LC 激光切割数控系统
 PNC2000SC 海绵切割数控系统
 PNC2000PC 等离子/火焰切割数控系统
 PNC2000LC 水射流切割数控系统
 PNC2000PS 六轴喷涂机械手数控系统
 PNC2000SM 开槽机数控系统
 PNC2000EM 倒角机数控系统

PNC2000SL 旋压机数控系统
PNC2000RB 钢筋弯曲数控系统
PNC2000LW 激光焊接数控系统
PNC2000IC 智能卡绕线数控系统
PNC2000CG 单晶炉控制系统