User Manual

Edition 07/2002
File MAE-PBstepper-E.xxx
Order Code: MAE-Pbstepper-E

PROFIBUS DP STEPPER 6411-PBX

for firmware version 5.1

Danaher Motion GmbH
Robert-Bosch-Str. 10
D-64331 Weiterstadt, Germany
Tel.: +49 (0)6151 - 8796 - 10
Fax: +49 (0)6151 - 8796 - 123
Email: info@danahermotion.net
Internet: www.danahermotion.net

BAUTZ

Trademarks used:

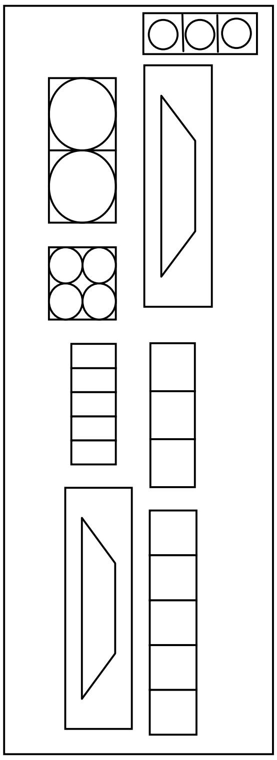
IBM and PC-AT are registered trademarks of International Business Machines Corp.
Simatic S5 is a registered trademark of Siemens AG.

Danaher Motion GmbH reserves the right to make changes that serve to improve the quality of the product described in this manual without prior notice. The authors will be pleased to receive any comments with regard to errors, contradictions or clarity.

Contents

1. General

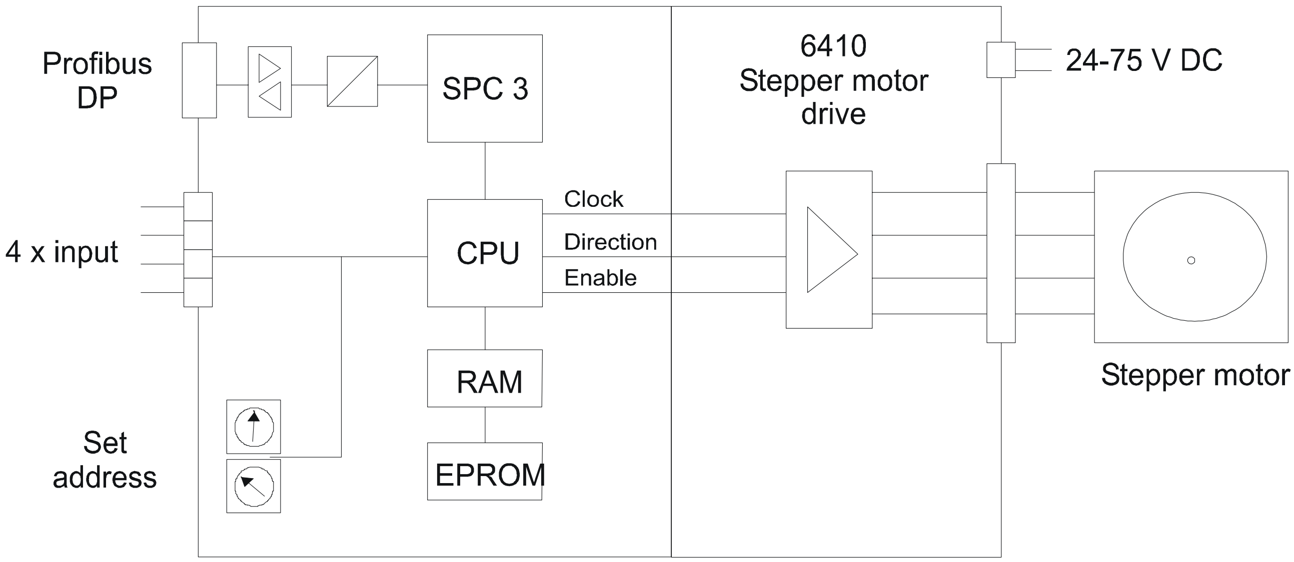
If you need to control several stepper motor drives in a distributed field application via PROFIBUS-DP, then the DP STEPPER is the answer. The DP STEPPER is a compact single-axis positioning control, with an integrated output stage for driving a stepper motor. It includes two limit switches, one stop switch (interrupt input) and one reference switch. Eight input and output bytes respectively are sufficient to control the STEPPER via PROFIBUS DP for handling single-axis positioning tasks. As the system only makes use of the process data channel, the STEPPER can be integrated into any control system that uses PROFIBUS DP as a sensor/actuator bus, without additional expenditure for installation. The fast, simultaneous transmission of the input and output bytes for all *devices* (participants) in the PROFIBUS DP system opens up numerous options for implementing multi-axis drives across the bus system, without having to put up with synchronization - problems.



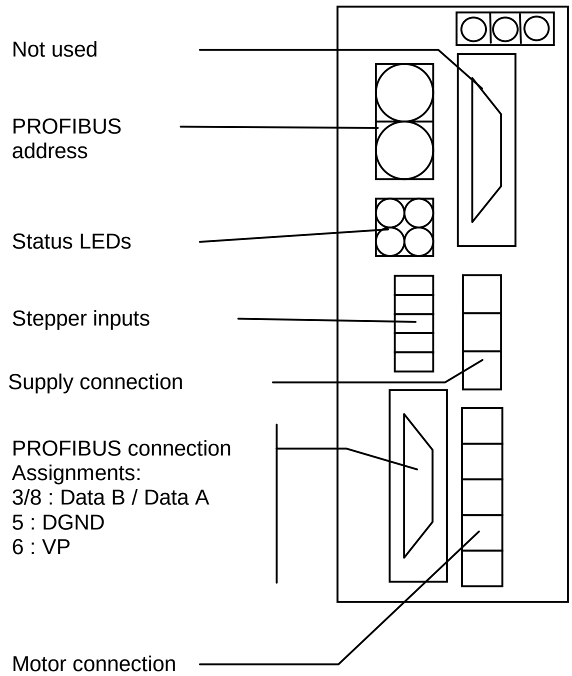
The drive cycle time can be flexibly adjusted by the control system, even while the drive is running. It is also possible to enter an acceleration up to a final limit or select a target position, again while the drive is running.
All status variables can be read in from the control system as two input bytes. Positioning tasks can therefore be implemented with the STEPPER in a very flexible way. And, for special applications, it is also possible to implement customer-specific functions in the STEPPER hardware in a fast, economic solution.

2. Hardware arrangement

The block diagram below shows the arrangement of the STEPPER hardware:

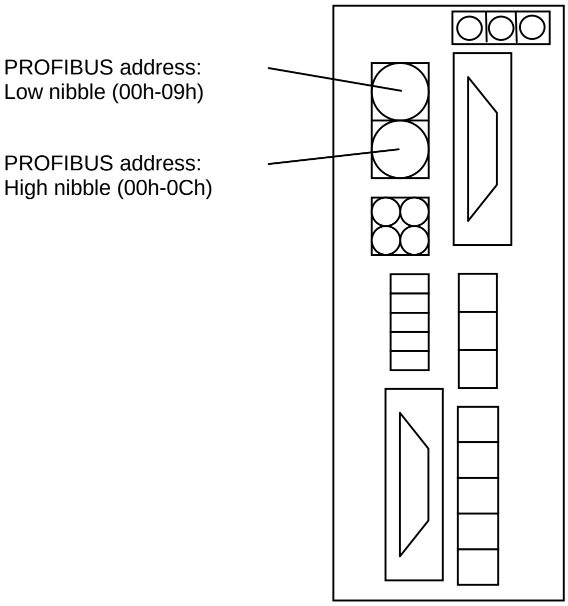


Position and function of connections



3. Configuration of the PROFIBUS DP address

The PROFIBUS address is set manually, using two rotary switches. The valid range for the PROFIBUS DP address is: 1-126

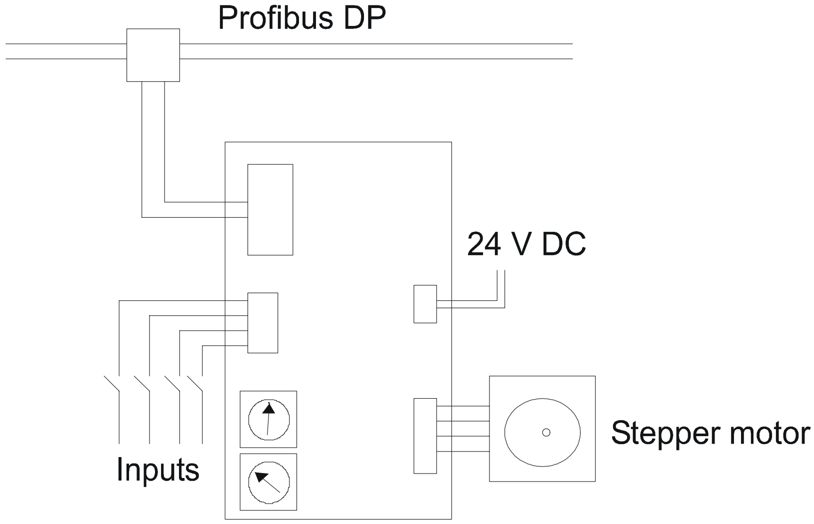


Examples of PROFIBUS addresses:

Address *High nibble*		**Address *Low nibble***		**PROFIBUS address**
Hex	**Decimal**	**Hex**	**Decimal**	**Decimal**
00	00	08	8	008
03	03	00	0	030
09	09	05	5	095
0A	10	03	3	103
0C	12	06	6	126

4. Handling

The DP Stepper is fitted with a fieldbus connection for PROFIBUS DP, and is integrated into PROFIBUS DP as a bus-system device. Inputs and outputs are electrically isolated from the control electronics, and run off an external 24 V DC supply. The homing (reference) run and all settings are completely set up via the PROFIBUS DP interface, and can be altered through the bus during operation. One output is assigned to the busy signal, so that electrical interlock functions can be implemented while the drive is running.



As seen by PROFIBUS DP, the STEPPER behaves as a device with **eight input bytes** and **twelve output bytes**. The control system can start or change drive actions by setting various bits in the **command word** (**output bytes 7 + 8**).
The control system can obtain the present **status** and actual position at any time, by simply reading the **input bytes 3 – 8**.
The absolute target position, to which the stepper should run after the next *Motor Start* command, is entered in output bytes 5 – 8. This target position can also be altered while the drive is moving. In this way, exact positioning can be performed without additional loading of the PROFIBUS DP master.

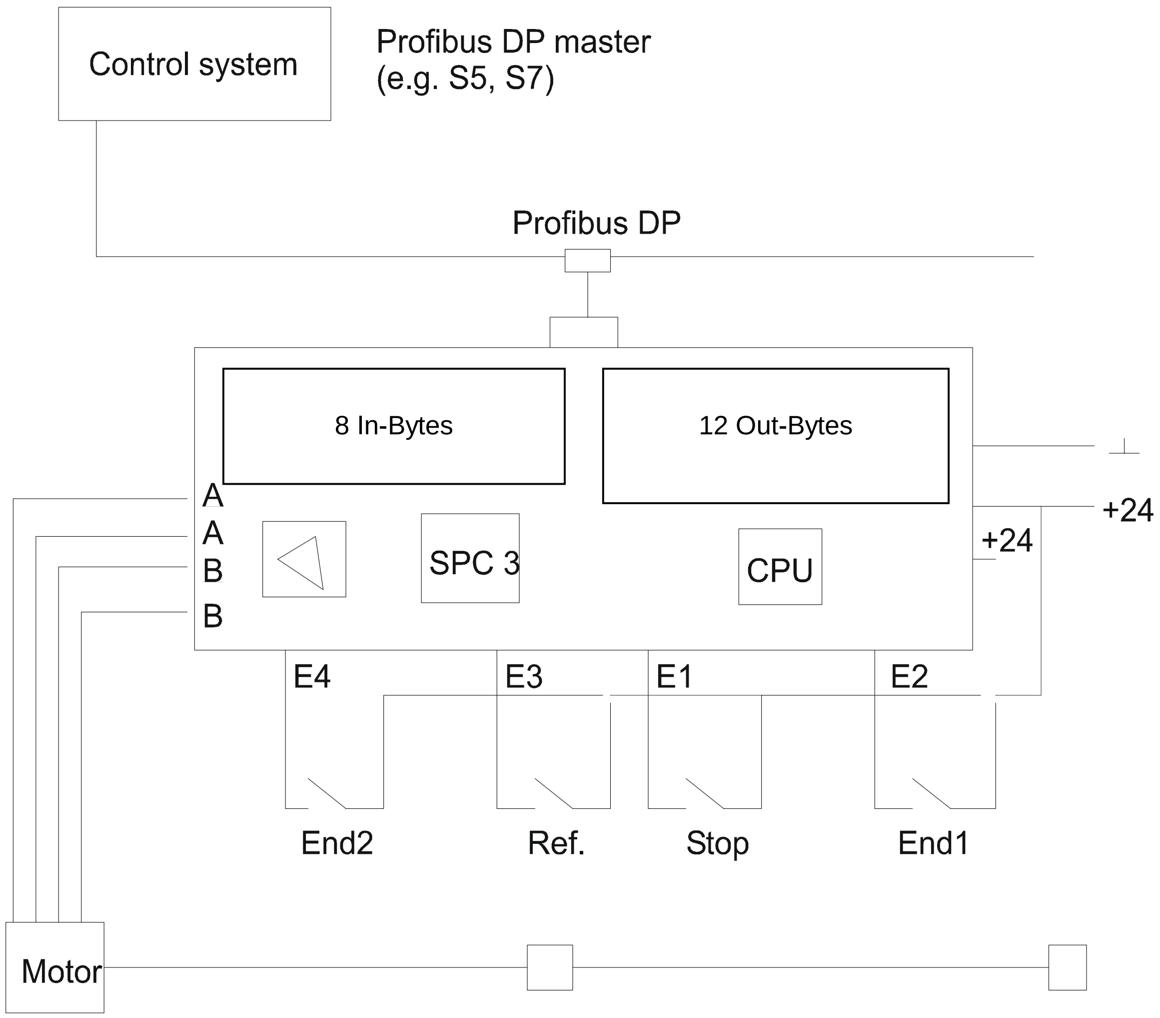
12 output bytes:

Byte no.	Meaning
1+2	Velocity code: Set velocity, for **normal movement.** Set target velocity, for **acceleration movement.**
3+4	Initial velocity (for acceleration movement)
5+6	Acceleration code, consisting of time unit and value division
7+8	Command word
9+10	Set target position, *Low word*
11+12	Set target position, *High word*

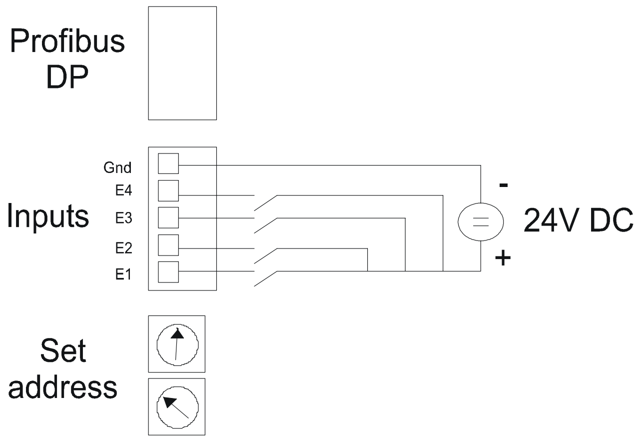
8 input bytes:

Byte no.	Meaning
1+2	Display the present velocity
3+4	Status bytes
5+6	Actual position, *Low word*
7+8	Actual position, *High word*

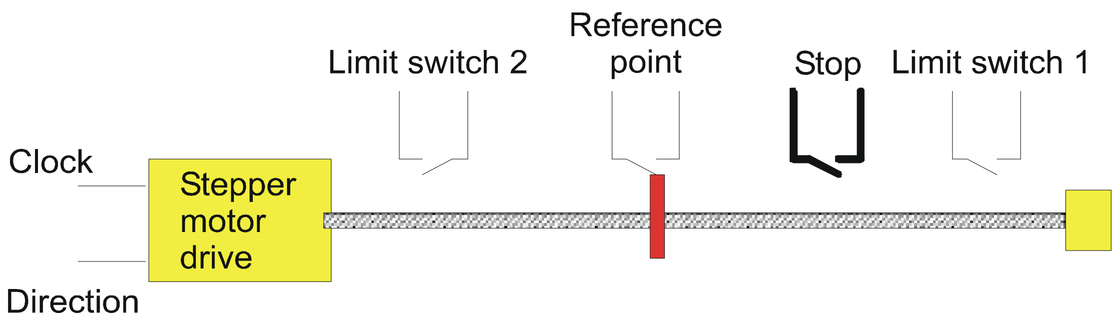
Connection of the inputs and outputs



Input connections



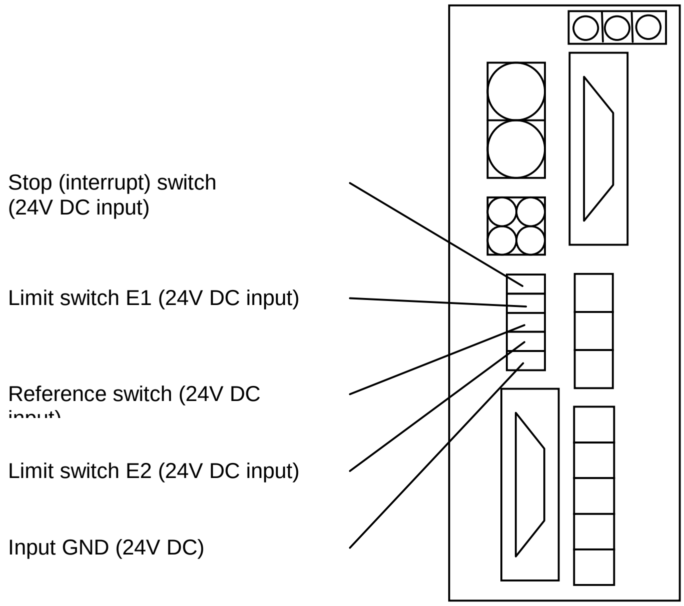
Schematic layout of a drive axis for the STEPPER



List of inputs and outputs

Upper limit switch (break contact) -------- Input 2
Lower limit switch (break contact) -------- Input 4
Reference point (make contact) ---------- Input 3
Stop (make or break contact)-------------- Input 1

Assignment of the inputs



5. Diagnosis LED and alarm status

A diagnosis LED is provided, to provide an at-a-glance check that the module processor is functioning correctly. The diagnosis LED flashes at a frequency of 5Hz.

! Alarm status !
As soon as the stepper switches on the alarm status, the diagnosis LED indicates this by lighting up continuously. The alarm status can be triggered by the following events:

1. The lower limit switch was activated during a homing run. (see section 8)

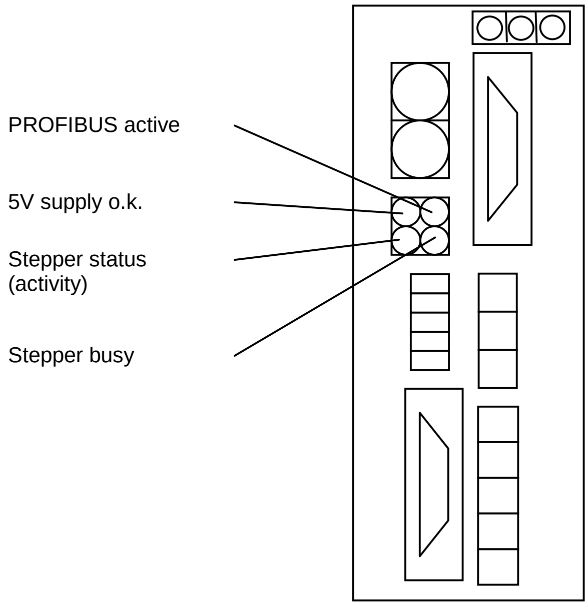
2. There is a processor fault.

No commands can be carried out while the alarm status is present. The stepper blocks off all PBS command information. However, PBS input information (i.e. status information and position signaling) will still be reported. The status information will now include the *Alarm status active* message.

! Terminate alarm status !
The stepper has been fitted with an option for terminating the alarm status without having to switch off the module. This is achieved by introducing a **pseudo-code** for the PBS output bytes 1+2 (velocity code). The code used is **0xAA55**. This code does not occur in normal operation, since the code for the maximum velocity is 0x1FF. If the stepper reads this pseudo-code, then it **checks again** whether the conditions for the alarm status are still present. If these conditions are no longer present, then it switches off the alarm status and returns to the operational condition **after** the pseudo-code has terminated. The diagnosis LED starts to flash again, indicating the operating condition. Please note that the output of the pseudo-code 0xAA55 must have really **stopped** in order for the normal operating condition to be resumed.

! Activate/deactivate control switches !
Entering the word 8001h in the velocity code activates the control switches. Entering 8000h deactivates the control switches.
Standard setting: Control switches activated.

Interpretation of the status LEDs



6. Programming the PBS stepper motor controller

The PBS stepper motor controller is controlled and monitored by eight input and eight output bytes, via PROFIBUS DP. The I/O bytes are assigned as follows:

12 output bytes

Byte no.	Meaning
1+2	Velocity code: Set velocity, for **normal movement.** Set target velocity, for **acceleration movement.** Byte 1: *High byte*; Byte 2 *Low byte*
3+4	Start velocity, for acceleration movement. Byte 3: *High byte; Byte 4 Low byte*
5+6	Acceleration code, consisting of time unit and value division Byte 5: *Time unit* ; Byte 6: *Value division*
7+8	Command word Byte7: *High byte*; Byte 8 *Low byte Low byte*
9+10	Set target position, *Low word* Byte 9: *High byte*; Byte 10 *Low byte*
11+12	Set target position, *High word* Byte 11: *High byte*; Byte 12 *Low byte*

The *Time step* in the acceleration code defines (how many) x 1 millisecond the drive should remain at a frequency step during the ramp time. The value division defines that one in every (value division) values from the table on Page 16 is to be used to generate the ramp(s). A detailed description of both of these parameters can be found on Page 14.

8 input bytes

Byte no.	Meaning
1+2	Display the present velocity Byte 1: *High byte*; Byte 2 *Low byte* Low byte
3+4	Status bytes Byte 3: *High byte*; Byte 4 *Low byte*
5+6	Actual position, *low word* Byte 5: *High byte*; Byte 6 *Low byte*
7+8	Actual position, *high word* Byte 7: *High byte*; Byte 8 *Low byte*

6.1 Structure of the command word and the acceleration code

In order to get the stepper motor to carry out an action, the control system must set the corresponding bits in the **Command word**. Nine bits are transmitted to the stepper motor controller for the transfer of commands. The bits in the command word have the following meaning:

Bits	Byte	Meaning	Range of values
00	8	Reserve	
01		Counter reset	0 = counter active 1 = counter reset (to 8000 0000h)
02		Acceleration on/off	0 = acceleration off (**normal movement**) 1 = acceleration on (**acceleration movement**)
03		Velocity mode	0 = position mode 1 = velocity mode
04		Motor start/stop	0 = stop motor/**initialize command** 1 = start motor
05		Direction of motion	0 = forwards 1 = backwards
06		Enable output	0 = off 1 = on
07		Homing	0 = normal mode 1 = homing (move to reference point)
08	7	Default rot. direction	0 = default direction of rotation as defined by Bit 05 1 = default direction of rotation, defined by inverted Bit 05
09		Polarity of stop switch	0 = normal / break contact 1 = inverse / make contact
10...15		Reserve	

Bit	Description
1	If this bit is set to 1, the step counter is reset to 8000 0000h in the input bytes 5-8. This bit must always be set to 0 for the normal positioning mode. The enable (Bit4) is ineffective as long as this bit is set
2	If this bit is set to 1, the acceleration mode of the stepper is activated, i.e. the selected final velocity is not reached instantly, but with some delay as a result of the acceleration that is defined in the acceleration code. If this bit is 0, the chosen step velocity is produced at once.
3	If this bit is set to 1, the stepper is operated in the velocity mode with an open-loop speed setpoint. No limit switches, stop switches or position setpoints will be evaluated. This bit must be set to 0 for normal positioning operation.
4	Setting this bit to 0 stops a motion command that is being performed. The transition from 0 to 1 enables the motor, and starts the corresponding motion command. When the motion command is finished, this bit must be set to 0 again.
5	This bit controls the direction of motion, i.e. the output direction of the stepper. The effect of Bit 8, which determines the default direction, must also be taken into account.
6	This bit is applied directly to the enable output of the stepper, and can be used to enable a group of stepper motors.
7	If this bit is set to 1, the normal mode of operation of the stepper is de-activated, and a homing run (movement to the reference point) is started. Homing is described in detail in Chapter 8.
8	This determines whether Bit 5 is evaluated normally or with inverse polarity. If Bit 8 = 0 then, for Bit 5, 0 = backwards and 1 = forwards. If Bit 8 = 1, then the opposite applies.
9	This bit determines the polarity of the stop input: 0 = break contact, 1 = make contact

Detailed make-up of the acceleration code

Bit15... Acceleration time step D7	Bit07... Acceleration value division D7
Bit14... Acceleration time step D6	Bit06... Acceleration value division D6
Bit13... Acceleration time step D5	Bit05... Acceleration value division D5
Bit12... Acceleration time step D4	Bit04... Acceleration value division D4
Bit11... Acceleration time step D3	Bit03... Acceleration value division D3
Bit10... Acceleration time step D2	Bit02... Acceleration value division D2
Bit09... Acceleration time step D1	Bit01... Acceleration value division D1
Bit08... Acceleration time step D0	Bit00... Acceleration value division D0

All details on the initial and final velocities in the examples refer to the coding as given in the table starting on Page 16:

<u>Example 1</u>

Given values: - initial velocity: 0
- final velocity: 535
- acceleration time step: 1
- acceleration value division: 10

Calculation: - internal index table:
0,10,20,30,40,50,60,70,80,90,100,110,120,130,140,150,160,170,180,
190,200,210,220,230,240,250,260,270,280,290,300,310,320,330,
340,350,360,370,380,390,400,410,420,430,440,450,460,470,480,
490,500,510,520, 520,530,535
- total time taken to reach the final velocity: 54 msec

<u>Example 2</u>

Given values: - initial velocity: 10
- final velocity: 50
- acceleration time step: 5
- acceleration value division: 1

Calculation: - internal index table:
10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,
31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50
- total time taken to reach the final velocity: 200 msec

<u>Example 3</u>

Given values: - initial velocity: 10
- final velocity: 535
- acceleration time step: 1
- acceleration value division: 52

Calculation: - internal index table:
10,63,116,169,222,275,328,381,434,487,535
- total time taken to reach the final velocity

6.2 Layout of the status word

In order to be able to transfer the present status of the stepper motor controller to the control system, the status bytes (**input bytes 3+4**) of the controller must be read. The table below shows how the bits within the status byte can be evaluated to determine the present status of the stepper controller.

Bits 0-7 = input byte 3, and Bits 8-15 = input byte 4 have the following meanings:

Bits	Byte	Meaning	Range of values
0	**4**	Stop switch status	0 = stop switch not activated 1 = stop switch activated
1		Lower limit switch status	0 = limit switch not activated 1 = limit switch activated
2		Reference point switch status	0 = reference point switch is open 1 = reference point switch is closed
3		Upper limit switch status	0 = limit switch not activated 1 = limit switch activated
4		Present direction of motion	0 = forwards (COUNT is incremented) 1 = backwards (COUNT is decremented)
5		Limit velocity status	0 = limit velocity not yet reached 1 = limit velocity reached
6		not assigned	
7		Busy indication	0 = step output is static (busy) 1 = step output is running
8	**3**	Limit switch status	0 = no limit switch reached 1 = a limit switch has been reached
9		Logic level of upper limit switch	
10		Logic level of lower limit switch	
11		Emergency stop / alarm	0 = all o.k. 1 = alarm or emergency stop activated
12		Counter status	0 = data are invalid 1 = counter data are valid
13		not assigned	
14		Module error	0 = module o.k. 1 = module error
15		not assigned	

6.3 Setting the value for the velocity code

The velocity code is contained in the **first two output bytes**. The possible **range of values** is from **000h to 32Bh**. The stepper uses an internally stored table to interpret the velocity code. This contains 811 integer values that encode the direct timer constants for the step clock rate.
The difference between one velocity value and the next is 15Hz for rates below 10kHz, and 30Hz for rates above 10 kHz. The nature of the timer structure in the processor means that the relationship is not perfectly linear. The exact values can be taken from the following table.
For special applications, it is possible to create a new code table and store it in the processor.
This table also applies for the initial velocity.

Code	Frequency (Hz)	Code	Frequency (Hz)	Code	Frequency (Hz)	Code	Frequency (Hz)	Code	Frequency (Hz)
0	30	35	825	6A	1620	9F	2415	D4	3222
1	45	36	840	6B	1635	A0	2430	D5	3237
2	60	37	855	6C	1650	A1	2445	D6	3252
3	75	38	870	6D	1665	A2	2460	D7	3267
4	90	39	885	6E	1680	A3	2475	D8	3282
5	105	3A	900	6F	1695	A4	2490	D9	3298
6	120	3B	915	70	1710	A5	2505	DA	3313
7	135	3C	930	71	1725	A6	2521	DB	3329
8	150	3D	945	72	1740	A7	2536	DC	3344
9	165	3E	960	73	1755	A8	2551	DD	3359
A	180	3F	975	74	1770	A9	2566	DE	3374
B	195	40	990	75	1785	AA	2581	DF	3389
C	210	41	1005	76	1800	AB	2596	E0	3405
D	225	42	1020	77	1815	AC	2612	E1	3420
E	240	43	1035	78	1830	AD	2628	E2	3436
F	255	44	1050	79	1845	AE	2643	E3	3452
10	270	45	1065	7A	1860	AF	2658	E4	3468
11	285	46	1080	7B	1875	B0	2673	E5	3484
12	300	47	1095	7C	1890	B1	2688	E6	3500
13	315	48	1110	7D	1905	B2	2703	E7	3516
14	330	49	1125	7E	1920	B3	2718	E8	3531
15	345	4A	1140	7F	1935	B4	2733	E9	3546
16	360	4B	1155	80	1950	B5	2748	EA	3562
17	375	4C	1170	81	1965	B6	2763	EB	3577
18	390	4D	1185	82	1980	B7	2779	EC	3592
19	405	4E	1200	83	1995	B8	2794	ED	3607
1A	420	4F	1215	84	2010	B9	2810	EE	3623
1B	435	50	1230	85	2025	BA	2826	EF	3638
1C	450	51	1245	86	2040	BB	2842	F0	3654
1D	465	52	1260	87	2055	BC	2857	F1	3669
1E	480	53	1275	88	2070	BD	2872	F2	3685
1F	495	54	1290	89	2085	BE	2887	F3	3701
20	510	55	1305	8A	2100	BF	2902	F4	3717
21	525	56	1320	8B	2115	C0	2918	F5	3733
22	540	57	1335	8C	2130	C1	2933	F6	3750
23	555	58	1350	8D	2145	C2	2948	F7	3766
24	570	59	1365	8E	2160	C3	2964	F8	3783
25	585	5A	1380	8F	2175	C4	2979	F9	3799
26	600	5B	1395	90	2190	C5	2994	FA	3814
27	615	5C	1410	91	2205	C6	3009	FB	3831
28	630	5D	1425	92	2220	C7	3024	FC	3846
29	645	5E	1440	93	2235	C8	3039	FD	3861
2A	660	5F	1455	94	2250	C9	3054	FE	3878
2B	675	60	1470	95	2265	CA	3069	FF	3893
2C	690	61	1485	96	2280	CB	3084	100	3908
2D	705	62	1500	97	2295	CC	3099	101	3924
2E	720	63	1515	98	2310	CD	3115	102	3939
2F	735	64	1530	99	2325	CE	3131	103	3955
30	750	65	1545	9A	2340	CF	3146	104	3970
31	765	66	1560	9B	2355	D0	3161	105	3986
32	780	67	1575	9C	2370	D1	3176	106	4002
33	795	68	1590	9D	2385	D2	3191	107	4018
34	810	69	1605	9E	2400	D3	3206	108	4034

Code	Frequency (Hz)
109	4051
10A	4067
10B	4084
10C	4101
10D	4118
10E	4135
10F	4152
110	4169
111	4184
112	4201
113	4216
114	4231
115	4246
116	4261
117	4276
118	4291
119	4307
11A	4322
11B	4338
11C	4354
11D	4369
11E	4385
11F	4402
120	4418
121	4434
122	4451
123	4467
124	4484
125	4501
126	4518
127	4535
128	4552
129	4569
12A	4587
12B	4604
12C	4622
12D	4640
12E	4658
12F	4676
130	4691
131	4709
132	4724
133	4739
134	4754
135	4769
136	4784
137	4800
138	4815
139	4830
13A	4846
13B	4862
13C	4878
13D	4893
13E	4909
13F	4926
140	4942
141	4958
142	4975
143	4991
144	5008
145	5025
146	5042
147	5059
148	5076
149	5093
14A	5110
14B	5128

Code	Frequency (Hz)
14C	5145
14D	5163
14E	5181
14F	5199
150	5217
151	5235
152	5253
153	5272
154	5291
155	5309
156	5328
157	5347
158	5366
159	5381
15A	5400
15B	5415
15C	5434
15D	5449
15E	5464
15F	5479
160	5494
161	5509
162	5524
163	5540
164	5555
165	5571
166	5586
167	5602
168	5617
169	5633
16A	5649
16B	5665
16C	5681
16D	5698
16E	5714
16F	5730
170	5747
171	5763
172	5780
173	5797
174	5813
175	5830
176	5847
177	5865
178	5882
179	5899
17A	5917
17B	5934
17C	5952
17D	5970
17E	5988
17F	6006
180	6024
181	6042
182	6060
183	6079
184	6097
185	6116
186	6134
187	6153
188	6172
189	6191
18A	6211
18B	6230
18C	6250
18D	6269
18E	6289

Code	Frequency (Hz)
18F	6309
190	6329
191	6349
192	6369
193	6389
194	6410
195	6430
196	6451
197	6472
198	6493
199	6514
19A	6535
19B	6550
19C	6571
19D	6586
19E	6607
19F	6622
1A0	6637
1A1	6659
1A2	6674
1A3	6696
1A4	6711
1A5	6726
1A6	6741
1A7	6756
1A8	6772
1A9	6787
1AA	6802
1AB	6818
1AC	6833
1AD	6849
1AE	6864
1AF	6880
1B0	6896
1B1	6912
1B2	6928
1B3	6944
1B4	6960
1B5	6976
1B6	6993
1B7	7009
1B8	7025
1B9	7042
1BA	7058
1BB	7075
1BC	7092
1BD	7109
1BE	7125
1BF	7142
1C0	7159
1C1	7177
1C2	7194
1C3	7211
1C4	7228
1C5	7246
1C6	7263
1C7	7281
1C8	7299
1C9	7317
1CA	7334
1CB	7352
1CC	7371
1CD	7389
1CE	7407
1CF	7425
1D0	7444
1D1	7462

Code	Frequency (Hz)
1D2	7481
1D3	7500
1D4	7518
1D5	7537
1D6	7556
1D7	7575
1D8	7594
1D9	7614
1DA	7633
1DB	7653
1DC	7672
1DD	7692
1DE	7712
1DF	7731
1E0	7751
1E1	7772
1E2	7792
1E3	7812
1E4	7832
1E5	7853
1E6	7874
1E7	7894
1E8	7915
1E9	7936
1EA	7957
1EB	7978
1EC	8000
1ED	8021
1EE	8042
1EF	8064
1F0	8086
1F1	8108
1F2	8130
1F3	8152
1F4	8174
1F5	8196
1F6	8219
1F7	8241
1F8	8264
1F9	8287
1FA	8310
1FB	8333
1FC	8356
1FD	8379
1FE	8403
1FF	8426
200	8450
201	8474
202	8498
203	8522
204	8547
205	8571
206	8595
207	8620
208	8645
209	8670
20A	8695
20B	8720
20C	8746
20D	8771
20E	8797
20F	8823
210	8849
211	8875
212	8902
213	8928
214	8955

Code	Frequency (Hz)
215	8982
216	9009
217	9036
218	9063
219	9090
21A	9118
21B	9146
21C	9174
21D	9202
21E	9230
21F	9259
220	9287
221	9302
222	9331
223	9360
224	9375
225	9404
226	9419
227	9448
228	9463
229	9478
22A	9493
22B	9508
22C	9523
22D	9538
22E	9554
22F	9569
230	9584
231	9600
232	9615
233	9630
234	9646
235	9661
236	9677
237	9693
238	9708
239	9724
23A	9740
23B	9756
23C	9771
23D	9787
23E	9803
23F	9819
240	9836
241	9852
242	9868
243	9884
244	9900
245	9917
246	9933
247	9950
248	9966
249	9983
24A	10000
24B	10016
24C	10050
24D	10084
24E	10118
24F	10152
250	10186
251	10221
252	10256
253	10291
254	10327
255	10362
256	10398
257	10434

Code	Frequency (Hz)
258	10471
259	10507
25A	10544
25B	10582
25C	10619
25D	10657
25E	10695
25F	10733
260	10771
261	10810
262	10849
263	10889
264	10928
265	10968
266	11009
267	11049
268	11090
269	11131
26A	11173
26B	11214
26C	11257
26D	11299
26E	11342
26F	11385
270	11428
271	11472
272	11516
273	11560
274	11605
275	11650
276	11695
277	11741
278	11787
279	11834
27A	11881
27B	11928
27C	11976
27D	12024
27E	12072
27F	12121
280	12170
281	12219
282	12269

Code	Frequency (Hz)
283	12320
284	12371
285	12422
286	12474
287	12526
288	12578
289	12631
28A	12684
28B	12738
28C	12793
28D	12847
28E	12903
28F	12958
290	13015
291	13071
292	13129
293	13186
294	13245
295	13303
296	13333
297	13363
298	13422
299	13452
29A	13483
29B	13513
29C	13544
29D	13574
29E	13605
29F	13636
2A0	13667
2A1	13698
2A2	13729
2A3	13761
2A4	13793
2A5	13824
2A6	13856
2A7	13888
2A8	13921
2A9	13953
2AA	13986
2AB	14018
2AC	14051
2AD	14084

Code	Frequency (Hz)
2AE	14117
2AF	14150
2B0	14184
2B1	14218
2B2	14251
2B3	14285
2B4	14319
2B5	14354
2B6	14388
2B7	14423
2B8	14457
2B9	14492
2BA	14527
2BB	14563
2BC	14598
2BD	14634
2BE	14669
2BF	14705
2C0	14742
2C1	14778
2C2	14814
2C3	14851
2C4	14888
2C5	14925
2C6	14962
2C7	15000
2C8	15037
2C9	15075
2CA	15113
2CB	15151
2CC	15189
2CD	15228
2CE	15267
2CF	15306
2D0	15345
2D1	15384
2D2	15424
2D3	15463
2D4	15503
2D5	15544
2D6	15584
2D7	15625
2D8	15665

Code	Frequency (Hz)
2D9	15706
2DA	15748
2DB	15789
2DC	15831
2DD	15873
2DE	15915
2DF	15957
2E0	16000
2E1	16042
2E2	16085
2E3	16129
2E4	16172
2E5	16216
2E6	16260
2E7	16304
2E8	16348
2E9	16393
2EA	16438
2EB	16483
2EC	16528
2ED	16574
2EE	16620
2EF	16666
2F0	16713
2F1	16759
2F2	16806
2F3	16853
2F4	16901
2F5	16949
2F6	16997
2F7	17045
2F8	17094
2F9	17142
2FA	17191
2FB	17241
2FC	17291
2FD	17341
2FE	17391
2FF	17441
300	17492
301	17543
302	17595
303	17647

Code	Frequency (Hz)
304	17699
305	17751
306	17804
307	17857
308	17910
309	17964
30A	18018
30B	18072
30C	18126
30D	18181
30E	18237
30F	18292
310	18348
311	18404
312	18461
313	18518
314	18575
315	18633
316	18691
317	18750
318	18808
319	18867
31A	18927
31B	18987
31C	19047
31D	19108
31E	19169
31F	19230
320	19292
321	19354
322	19417
323	19480
324	19543
325	19607
326	19672
327	19736
328	19801
329	19867
32A	19933
32B	20000

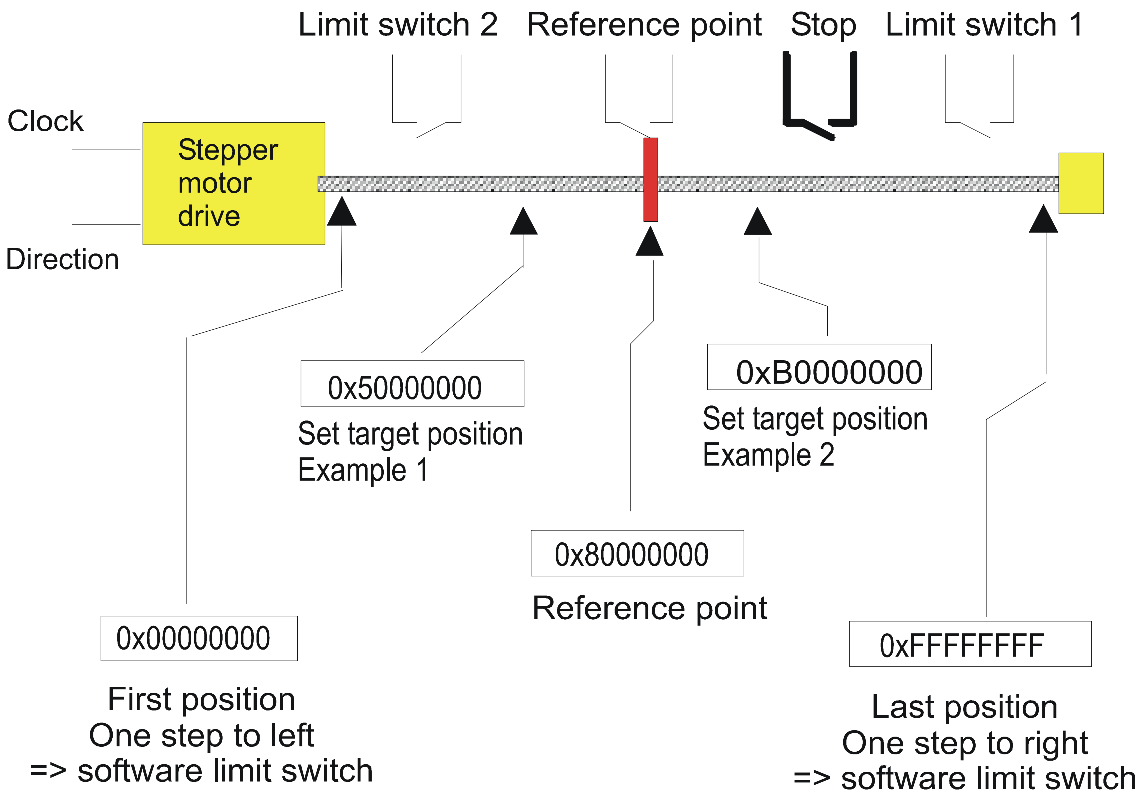
6.4 Positioning

The stepper uses **absolute** positioning. This clear and unambiguous type of positioning is feasible because the range of values for the counter is an unsigned long variable. As a consequence, a count of **0xFFFFFFFF steps** is available for the range of values. Absolute positioning means that the count value 0 corresponds to position 0, and the count value 0xFFFFFFFF corresponds to position 0xFFFFFFFF.

The lower or upper software limit switch, respectively, will be activated if the count goes one step outside these two limits, and will then switch off the drive. An exception is made for the **velocity mode**, in which neither hardware nor software limit switches are used. If a software limit switch is reached in **normal mode** an error is indicated, since the actual physical limits for a real application are marked by the hardware limit switches.
The fundamental orientation for absolute positioning operation is provided by the reference point. The **reference point** is coded as position **0x80000000**. This value was chosen so that an adequate range of values is available for positioning in the ranges to both the left and right of the reference point. It is easy to convert the position value to suit the numerical format applied by the user. The use of an unsigned long value is a favorable starting point for conversion into various customer-specific numerical formats. This is the principle that is also used for analog input devices in the world of PROFIBUS DP and other fieldbuses.

The **actual position** of the stepper is given in **PBS input bytes 5-8**. The **setpoint position** (target position) is entered in **PBS output bytes 9-12**. As already explained above, the counter is set to 0x80000000 when the reference point is reached during the homing movement. Since the position is indefinite **after switch-on** of the equipment, the counter is also **set to 0x80000000** in these circumstances, so that the user has the possibility of moving the drive in both directions without immediately running onto a software limit switch. For example, if the counter was set to 0 by a reset, then a backwards movement would instantly run into the lower software limit switch. The **Counter Reset** command has been implemented in such a manner that the counter is set to the position value 0x80000000, for the same reason.

Reading out the present state of the counter enables the control system to derive the momentary position of the drive and to evaluate this within the application program.
It is not necessary for the control system to read out the present counter value for positioning, since the set target position is provided in the output bytes 9-12. The module compares the actual position with the target position at every step, and stops the stepping procedure as soon as both positions are the same. The target position will not be overrun by a single step. This method avoids positioning errors caused by response times.
If the stepper has stopped generating steps, because it has reached the target position, then the application program can select a new target position. However, before the stepper can start moving **again**, the **motor start bit** of the **command bytes**, that may still be active from the preceding movement, **must first be set to be inactive** and then **set to be active again**. This extra security measure is valuable, since the stepper might otherwise start immediately if a value was still present in the lower-value section of the target position value, even though the higher-value section had not yet been altered. The target position can also be altered **during** the movement. If this function is to be used within the application program, then care must be taken that the target position is altered **between two PBS data cycles**, so that these can be transmitted with internal consistency, since the drive carries on running during the alteration.
It is also necessary to note that the stepper will switch the direction output by itself if the acceleration mode is inactive. The corresponding bit in the command will therefore be ineffective. The direction of movement selected by the stepper is always the result of the comparison between the actual and target positions. The command bit for direction of motion is only effective in velocity mode, since there is no comparison of positions in this mode.



6.5 Use of the target position

The **stop conditions** are checked continuously during the movement. Hardware conditions (stop, limit, and emergency stop switches) naturally have a higher priority, i.e. the motor will stop immediately if the target position has not yet been reached, but one of these switches is activated.

The set target position is not evaluated during a **homing** movement, since homing is controlled by the stepper itself.

The target position is also not evaluated in **velocity mode**.

In normal mode, the set target position can be altered during the movement. However, in this case it must be observed that the target position is altered in a consistent manner, since the stepper instantly takes the position value that it has received and uses it for the position comparison. The required data consistency is achieved by altering the position bytes between two PBS data cycles. If the hardware setup of the master that is used makes this impossible, then position alteration during the movement should not be used. A position value that is not transmitted in a consistent manner may cause the stepper to stop unintentionally, or, with unfavorable position values, even change the direction of motion.
Since the stepper will stop when the target position has been reached, the next target position can be selected without any problem of consistency. The same applies to stops caused by a limit switch or cancelation of the motor enable signal. In these cases, the entry of the set target position can be spread over several PBS data cycles, if the motor start command is only activated again for the next movement after the complete set target position has been entered.

For stops on a limit switch, alarm stops, or stops on reaching the target position, the **next motor start** can **only** be made when the **Motor start bit** has first been **reset** and then **set again**. This ensures complete data consistency after a stop.

7. Command examples

Command example 1   medium speed backwards

After output of the command described below, the stepper first moves from the actual position 8000 0000h with the clock frequency 495Hz to the target 7FFF0BB8h and remains stopped in that position. The initial frequency is to be 0Hz. After switching on the equipment, or a counter reset, or a successful homing movement, the counter state for the actual position is 8000 0000 h. Care must be taken when setting the PBS-OUT byte, that the motor-start bit is enabled last.

Frequency/ clock rate	01Fh -> 495Hz
Actual position	8000 0000 h
Target position	7FFF 0BB8 h
Stop	break contact (inactive)
Upper limit switch	closed (inactive)
Lower limit switch	closed (inactive)
Direction of motion	backwards
Operating mode	normal operation
Motor enable	yes (motor is enabled)

This leads to the following assignments for the PBS-OUT bytes:

Velocity code	PBS-OUT bytes 1+2	001Fh	0000 0000 0001 1111
Initial velocity code	PBS-OUT bytes 3+4	0000h	0000 0000 0000 0000
Acceleration code	PBS-OUT bytes 5+6	0000h	0000 0000 0000 0000
Target position, Low word	PBS-OUT bytes 9+10	0BB8h	0000 1011 1011 1000
Target position, High word	PBS-OUT bytes 11+12	7FFFh	0111 1111 1111 1111

When the task is finished, the PBS-IN bytes will have the following assignments:

Actual velocity	PBS-IN bytes 1+2	001Fh	0000 0000 0001 1111
Status bytes	PBS-IN bytes 3+4	0010h	0000 0000 0111 0000
Low word for position value	PBS-IN bytes 5+6	0BB8h	0000 1011 1011 1000
High word for position value	PBS-IN bytes 7+8	7FFFh	0111 1111 1111 1111

Command example 2 Acceleration movement, forwards

After output of the command described below, the stepper moves forwards in acceleration mode from the actual position 8000 0000h and the initial velocity 0. The acceleration time is 147 msec and the final velocity (frequency) is 16 kHz. The drive stops when it has reached the target position. After switching on the equipment, or a counter reset, or a successful homing movement, the counter state for the actual position is 8000 0000 h. Care must be taken when setting the OUT byte, that the motor-start bit is enabled last.

Final velocity (i.e. clock rate/frequency)	02E0h -> 16000Hz
Actual position	8000 0000 h
Target position	8001 0000 h
Stop	break contact (inactive)
Upper limit switch	closed (inactive)
Lower limit switch	closed (inactive)
Direction of motion	forwards
Operating mode	acceleration movement
Acceleration code	0105 -> time step 1msec, using every fifth value from the table
Motor enable	yes (motor is enabled)

This leads to the following assignments for the PBS-OUT bytes:

Velocity code	PBS-OUT bytes 1+2	02E0h	0000 0010 1110 0000
Initial velocity	PBS-OUT bytes 1+2	0000h	0000 0000 0000 0000
Acceleration code	PBS-OUT bytes 5+6	0105h	0000 0001 0000 0101
Command word	PBS-OUT bytes 7+8	0054h	0000 0000 0101 0100
Target position Low	PBS-OUT bytes 9+10	0000h	0000 0000 0000 0000
Target position High	PBS-OUT bytes 11+12	8001h	1000 0000 0000 0001

When the task is finished, the PBS-IN bytes will have the following assignments:

Actual velocity (frequency)	PBS-IN bytes 1+2	0000h	0000 0000 0000 0000
Status bytes	PBS-IN bytes 3+4	0000h	0000 0000 0000 0000
Low word for position value	PBS-IN bytes 5+6	0000h	0000 0000 0000 0000
High word for position value	PBS-IN bytes 7+8	8001h	1000 0000 0000 0001

<u>Command example 3</u> Clock output in velocity mode

Velocity mode is a special feature within the STEPPER functions. In this mode the module functions as a speed control that can be adjusted via PROFIBUS DP to match the predetermined velocity profile. Hardware and software limit switches, the stop switch and the target position are all not evaluated in this mode. However, the emergency stop switch (both limit switches activated simultaneously) is evaluated, and if it is activated the drive will be stopped at once. Acceleration movements are possible, and the momentary counter state is reported. When setting the OUT bytes, care must be taken that the motor start bit is the last one to be enabled.

Clock frequency	3Fh -> 975Hz
Actual position	8000 0000 h
Target position	xxxx xxxx h (not evaluated)
Stop	break contact (inactive)
Upper limit switch	x (not evaluated)
Lower limit switch	x (not evaluated)
Direction of motion	forwards
Operating mode	velocity mode, without acceleration
Motor enable	yes (motor is enabled)

This leads to the following assignments for the PBS-OUT bytes:

Velocity code	PBS-OUT bytes 1+2	003Fh	0000 0000 0011 1111
Initial velocity	PBS-OUT bytes 3+4	xxxxh	xxxx xxxx xxxx xxxx
Acceleration code	PBS-OUT bytes 5+6	xxxxh	xxxx xxxx xxxx xxxx
Command word	PBS-OUT bytes 7+8	0048h	0000 0000 0100 1000
Target position, Low word	PBS-OUT bytes 5+6	xxxxh	xxxx xxxx xxxx xxxx
Target position, High word	PBS-OUT bytes 7+8	xxxxh	xxxx xxxx xxxx xxxx

When the task is finished, the PBS-IN bytes will have the following assignments:

Actual velocity (frequency)	PBS-IN bytes 1+2	003Fh	0000 0000 0011 1111
Status bytes	PBS-IN bytes 3+4	00A0h	0000 0000 1010 0000
Low word for position value	PBS-IN bytes 5+6	xxxxh	xxxx xxxx xxxx xxxx
High word for position value	PBS-IN bytes 7+8	xxxxh	xxxx xxxx xxxx xxxx

Example Software stop in forwards motion

Frequency/ clock rate	03
Stop	make contact (inactive)
Upper limit switch	closed (inactive)
Lower limit switch	closed (inactive)
Direction of motion	forwards
Operating mode	normal operation
Motor enable	no (motor should stop)

This leads to the following assignments for the PBS-OUT bytes:

Velocity code	PBS-OUT bytes 1+2	0003h	0000 0000 0000 0011
Command word	PBS-OUT bytes 7+8	0040h	0000 0000 0100 0000
Target position, Low word	PBS-OUT bytes 5+6	1000h	0001 0000 0000 0000
Target position, High word	PBS-OUT bytes 7+8	8FFFh	1000 1111 1111 1111

When the command has been given, the motor stops and the assignments for the PBS-IN bytes are as follows:

Actual velocity (frequency)	PBS-IN bytes 1+2	0000h	0000 0000 0000 0000
Status bytes	PBS-IN bytes 3+4	0000h	0000 0000 0000 0000
Actual position, Low word	PBS-IN bytes 5+6	xxxxh	xxxx xxxx xxxx xxxx
Actual position, High word	PBS-IN bytes 7+8	xxxxh	xxxx xxxx xxxx xxxx

Other conditions recognized for a motor stop:

Stop switch close, with stop:	Status bytes1+2 = 0x1001
Upper limit switch opens, with stop:	Status bytes1+2 = 0x1308
Counter overflow in forwards motion, with stop	Status bytes1+2 = 0x1300

If a hardware or software limit switch is activated, it is only possible to leave the limit switch position by moving in the opposite direction.

8. Start Homing

The homing movement (to the reference point) is a special feature within the commands for the stepper motor controller. When this command is carried out, the velocity (frequency) is fixed at the value 0x20. The homing movement is always started by setting the command bit 7 (PC).

There are two options for performing the homing movement:

- Homing movement starting in the **backwards** direction
- Homing movement starting in the **forwards** direction

The choice must be made by a corresponding setting of the direction bit (Bit 5) in the command byte.

The logic ensures that both homing movements terminate in the same sense for the reference point switch. If the direction bit is reversed during one of the two possible homing movements, then an **alarm status** is produced.

If the reference point switch is not detected between the upper and lower limit switches, then the module also goes into the alarm status, since there must be a **hardware error** present. Please note that the execution of other commands is blocked until the homing movement has been completed successfully.

When the homing movement has been completed successfully, the drive is at standstill exactly at the reference point, and the position counter has the value 0x80000000h. The controller is now ready to accept commands and continue with normal operation. Homing can be initiated by the control system at any time. The homing movement can only be interrupted by an emergency stop, the stop switch, or by reaching the lower limit switch, since in the last case it can be assumed that the reference switch was not detected. If an emergency stop is activated or the lower limit switch is reached, the alarm status is produced, but if the stop switch is activated during homing the drive will simply interrupt the homing movement and stop. The control system can read all status conditions from the status bytes and then initiate the appropriate action.
When the homing movement has been completed successfully, the drive stands exactly at the reference point, the position counter has the value 0x80000000h, and the drive is ready to move forwards or backwards – this is the basic/initial position. The status bytes indicate that the data from the position counter are valid from now on.

The following summary serves to clarify the switching sequence during a homing movement.

Abbreviations
VC = velocity code
LS = limit switch
RS = reference point switch

1. **Forwards homing**, slide is between the reference point and the upper limit switch:

<table>
<tr><td>1.1</td><td colspan="3">Position counter is incremented, VC as selected.
Forwards homing movement starts.</td></tr>
<tr><td>1.2</td><td colspan="3">Lower LS activated. Lower LS is permitted to be active at the start.</td></tr>
<tr><td>1.3</td><td colspan="3">Upper LS activated. Drive changes direction, position counter decremented.
VC as selected. Reference point is below the starting position.</td></tr>
<tr><td>1.4</td><td>a) RS activated.
Drive changes direction.
Counter is incremented,
VC = 5
RS detected from higher position, back slowly until RS is free again.</td><td>b) Lower LS activated.
Alarm status !
RS is missing.</td><td>c) Upper LS activated. This is permissible after the change of direction.</td></tr>
<tr><td>1.5</td><td colspan="3">RS changes to inactive. Drive changes direction, position counter decremented, VC = 5, creep up to RS.</td></tr>
<tr><td>1.6</td><td>a) RS activated, position counter stands at 8000 0000h, ref. point has been reached.</td><td>b) Lower LS activated.
Alarm status !
RS is jammed.</td><td>Upper LS activated. Movement now in decrement direction.</td></tr>
</table>

2. **Forwards homing**, slide is between the reference point and the lower limit switch, explanations analogous to example 1.

2.1	Start, position counter is incremented, VC as selected.		
2.2	RS activated, no change.		
2.3	RS changes to inactive, drive changes direction, position counter decremented, VC=5.		
2.4	a) RS activated, position counter stands at 8000 0000h, ref. point has been reached.	b) Lower LS activated. Alarm status !	c) Upper LS activated, no change.

3. **Forwards homing**, direction changed to backwards during the movement.

3.1	Start, position counter is incremented, VC as selected, movement begins.
3.2	Direction changed to backwards, alarm status ! – The direction must not be changed during the movement.

4. **Backwards homing**, slide is between the reference point and the lower limit switch, explanations analogous to example 1.

4.1	Position counter is decremented, VC as selected. Homing movement starts in the **backwards** direction		
4.2	Upper LS activated, no change.		
4.3	Lower LS activated. Drive changes direction, position counter incremented. VC as selected.		
4.4	RS activated, no change.		
4.5	RS changes to inactive, drive changes direction, position counter decremented, VC=5.		
4.6	a) RS activated. Position counter stands at 8000 0000h, ref. point has been reached.	b) Upper LS activated, no change.	c) Lower LS activated, alarm status !

3. **Backwards homing**, slide is between the reference point and the upper limit switch:

5.1	Position counter is decremented, VC as selected. Homing movement starts in the **backwards** direction		
5.2	a) RS activated. Drive changes direction. Counter is incremented, VC = 5	b) Upper LS activated, alarm status !	c) Lower LS activated, no change.
5.3	RS changes to inactive. Drive changes direction, position counter decremented, VC = 5, creep up to RS.		
5.4	a) RS activated, position counter stands at 8000 0000h, ref. point has been reached.	b) Lower LS activated, alarm status !	c) Upper LS activated, no change.

6. **Backwards homing**, direction changed to forwards during the movement.

6.1	Start, position counter is decremented, VC as selected, movement begins.
6.2	Direction changed to forwards, alarm status ! – The direction must not be changed during the movement.

Fault conditions
• As soon as the reference switch has been reached for the first time, all subsequent forwards movements will evaluate the upper limit switch as a reference fault, and all subsequent backwards movements will evaluate the lower limit switch as a reference fault. • As soon as the upper limit switch has caused a change of direction to backwards movement, the lower limit switch will be evaluated as a reference fault.

General comments
• The software limit switches are suppressed during a homing movement. The homing movement can therefore cover the entire range of values. • During homing there is no comparison of the position with the target position. • During homing, the EN output is set as defined in the corresponding command byte. • The status bit *Reached final velocity* is set during homing, as long as the drive is moving at the velocity set by the velocity code. • A **new homing movement** can only be started after the command **Counter Reset** has been given, or the equipment is switched on again.

Example of homing movement starting in the **forwards** direction:

Assignment of the PBS-OUT bytes:

Velocity code	PBS-OUT bytes 1+2	0005h	0000 0000 0000 0101
Command word	PBS-OUT bytes 7+8	00C0h	0000 0000 1010 0000
Target position, Low word	PBS-OUT bytes 5+6	xxxxh	xxxx xxxx xxxx xxxx
Target position, High word	PBS-OUT bytes 7+8	xxxxh	xxxx xxxx xxxx xxxx

When the command has been generated, the assignments for the PBS-IN bytes are as follows:

Actual velocity (frequency)	PBS-IN bytes 1+2	0005h	0000 0000 0000 0101
Status bytes	PBS-IN bytes 3+4	1000h	0001 0000 0100 0000
Actual position, Low word	PBS-IN bytes 5+6	0000h	0000 0000 0000 0000
Actual position, High word	PBS-IN bytes 7+8	8000h	1000 0000 0000 0000

9. Emergency stop function

The STEPPER is provided with the option of activating a manual emergency stop in the event of a fault or a dangerous situation. In the event of an emergency stop, the drive stops immediately and the STEPPER will no longer accept any commands. This does not, however, interrupt the operation of PROFIBUS DP, so that the other devices in the PROFIBUS DP system can continue to operate.

The emergency stop function does not have a special input, but is triggered by simultaneous activation of both limit switches. Since the limit switches are implemented as break contacts, to achieve a fail-safe response if a cable break occurs, the emergency stop function can be implemented by wiring the emergency stop switch so that, if it is activated, the 24V signal is simultaneously disconnected from inputs E2 and E4.

Note

It must be ensured that the 24V signal is already applied to the limit switch inputs before the STEPPER starts, otherwise the emergency stop will be accidentally activated and the STEPPER will be unable to move.

Attachment A: Housing dimensions