KOLLMORGEN®

Description of S7 Handling Function FB10 for S300, S400, S600, S700

All cross references to the manual refer to the technical description PROFIBUS S300/S400/S600/S700 Communication Profile, Edition 12/2010. Handling of the FB10 is the same for all servo drives from the S300, S400, S600 and S700 series. In the following text servo drive does always mean these drives.

The function block package consists of the following components:

FB10	Handling function
DB110	Instance data block for FB10
DB101	User data DB

All function -/ data – blocks can be renamed if needed.
The FB10 uses the S7 system functions SFC14 (Receive) and SFC15 (Send) to communicate via the Profibus. The data will be stored in the user data block and transmitted consistently from there.

Pre-Conditions

For each drive one user data block is required. FB10 must be called with an individual instance data block for each servo drive.

Structure of the User Data Block

Address	**Name**	**Comment**
0	KSTW	Beginning of user data
... 56		Send and Receive Data
58	Jog_Speed	Speed for jog mode [1)]
60	Ref_Speed	Speed for reference move [1)]
62	Motion_Task_No	Number of motion task to be started (0 = direct motion task)
64	Direct_MT_type	Type of direct motion task
66	Direct_vtarget	Speed command of direct motion task
70	Direct_starget	Position command of direct motion task
74	DW_74	Reserved
76	Obj_Parameter_read	Result of successful reading operation
80	Operation_Profile	Operation Mode set with Init_Drive (according to Profidrive Profile)
82	User_Mode	Bitvariable, meaning see below
84	Obj_No	Number of parameter to be changed
86	Obj_Parameter_write	Value of parameter to be changed

[1)] For reference speed and jog speed the resolution set with the PC setup software is taken into account. The speed results from the value in the DB multiplied with a multiplier (PNU 1894, ASCII parameter VMUL). The default value of the multiplier is 1.

The memory up to address 56 is used to send and receive data and must not be changed by the user. Parameters beginnig with address 58 can be changed by the user as described below.

Inputs and Outputs of the FB10

Inputs

Name	Type	Comment
NUDB	Integer	Number of the user data block
Bus_Address	Integer	Profibus Slave Address
PERI_Address	Integer	Periphery Address (Address of 4AX in the hardware configuration)
SW_Enable	Bool	Software enable
Reset_Fault	Bool	Fault reset
Jog_for	Bool	Jog forward

Jog_back	Bool	Jog backward
Ref_Start	Bool	Start reference move
Start_MT	Bool	Start motion task (depending on bit 0 in User Mode, s.below)
Fast_Stop_Disable	Bool	1 ->0 Fast stop with disable. To reenable the drive after a fast stop the input SW_Enable must be reset and set again.
Fast_Stop_Enable	Bool	1 -> 0 Fast stop (Drive stays enabled)
Pause	Bool	0 -> 1 Interrupt motion task, continue with 1 -> 0
Position_Reset	Bool	Reset Position (actual position = reference offset ROFFS)
Accept_Values	Bool	Write value from user DB to drive. PNU/ object number specified in user DB; ACK_Ready is set when ready
Init_Drive	Bool	Initilize drive, set operation mode (PNU 930) to value specified in UserDB.DW80; ACK_Ready is set when ready

Outputs

Name	Type	Comment
Actual_Position	DWord	Actual position in increments [1)]
Actual_Speed	Word	compare manual description of operation mode 2
Manufac_Stat	Word	manufacturer specific status
ZSW	Word	status word
Error	Bool	error on drive
Pos_Error	Bool	position following error
Moving	Bool	speed <> 0, adjustable with parameter VEL0
Reference_ok	Bool	reference point set
In_Position	Bool	in position window reached
Ack_ready_ok	Bool	writing or reading of parameter finished successfully
Ack_ready_fault	Bool	writing or reading of parameter finished with error
MT_active	Bool	motion task active

[1)] In the process data channel the actual position is only transferred in the servo drive internal units (2^{20} Incr. per motor rev.). The calculation to get user units can be done in the PLC according to the resolution set with the setup software.
Example: Resolution = 5000 µm / 3 revs
=> Position in user units = Actual_Position x 5000 / (3 x 2^{20})

Commissioning

Initialization

- Write required operation mode (compare manual) into UserDB.DBW 80 “Operation_Profile”, e.g. 2 for “Positioning”.
- Set Init_Drive.
- Wait until Ack_Ready_ok = 1 or Ack_Ready_fault = 1.
- When Ack_Ready_fault = 1, look for reason for not successful initilization.
- Set Init_Drive back.
- Ack_Ready_ok/fault will be set back.
- Set Fast_Stop_Disable and Fast_Stop_Enable = 1. (They must always be 1 and only be reset for Fast Stop.)

Start Reference Move

The type of reference move is to be set with the setup software or through PNU 1773 (s. manual chap. 4.2.6.1)

- Set SW_Enable.
- Write Ref_Speed in the user DB (s. also description of the DB above).
- Set Ref_Start.
- Wait until MT_active = 1.
- Wait until MT_active = 0 and Ref_ok = 1.
- Set Ref_Start back.

Reference move has been finished.

Start an EEPROM Motion Task (which has been defined before with the setup software)

- Write number of motion task to be started into UserDB.DBW62 “Motion_Task_No”.
- Set Start_MT.
- Wait until In_Position = 0.
- Wait, until In_Position = 1 and MT_active = 0.
- Set Start_MT back.

Start a Direct Motion Task

- Write 0 into UserDB.DBW62 “Motion_Task_No”.
- Write speed command into UserDB.DBD 66 “Direct_vtarget”,.
- Write target position into UserDB.DBD 70 “Direct_starget”
- Write motion task type into UserDB.DBW 64 “Direct_MT_type”.
- Set Start_MT.
- Wait, until In_Position = 0.
- Wait, until In_Position = 1 and MT_active = 0.
- Set Start_MT back.

Note: With the parameter INPT you can set a time delay in ms for the In_Position signal. After the start of a motion task the signal will be set to 0 at least for the time INPT (s. manual, chap. 2.2.2).

Note for position and speed command: Target position and speed command can be set in user units, if bit 13 of the motion task type is set.

Motion Task Types (common values)
(s. manual, chap. 4.2.5.3)
0x2000 (Bit 13 set) Absolute positioning; speed and position setpoint in user units.
0x2003 (Bits 0, 1, 13 set) Positioning “relative command”; position and speed setpoint in user units.

Jog Mode

- Write jog speed to UserDB.DBW 58 “Jog_Speed”
- Set Jog_for or Jog_back
- The drive will run forward or backward as long as the input is active

If both inputs are active at the same time, the drive stops.

Read and Write Parameters
All parameters and commands of the servo drive can be accessed via their object number or a PNU resulting from this. The ASCII command list can show all parameters listed by the object number. Object numbers and PNUs can also be found with the descrition of each command/ parameter in the fields “DPR” and Profibus PNU. In addition all PNUs described in the manual can be acessed.
Bit 2 of the variable User_Mode, UserDB.DBX 83.2, determines if the number is a PNU (Bit 2 = 1) or an object number (Bit 2 = 0).

Write a Parameter

- Set bit 1 of the variable User_Mode, UserDB.DBX 83.1, equal to “1”
- Write object number/PNU to UserDB.DBW 84 “Obj_No”.
- Write value to the UserDB.DBD 86.
- Set Accept_Value.
- Wait, until Ack_Ready_ok = 1, or Ack_Ready_fault = 1
- If Ack_Ready_fault = 1, check object no. and value range
- Reset Accept_Value
- Ack_Ready_ok resp. Ack_Ready_fault will be set back

Read a Parameter

- Set bit 1 in the variable User_Mode, UserDB.DBX 83.1, equal to “0”
- Write object number/ PNU to UserDB.DBW 84 “Obj_No”
- Set Accept_Value.
- Wait, until Ack_Ready_ok = 1 or Ack_Ready_fault = 1.
- If Ack_Ready_fault = 1, check object number.
- If Ack_Ready_ok = 1, find the data in UserDB.DBD 76 “OBJ_Parameter_read”.
- Set Accept_Value back.
- Ack_Ready_ok resp. Ack_Ready_fault is set back.

Variable User_Mode

DBW82	Axis.User_Mode	Word	
DBX83.0	Bit 0	0	A motion task is started by setting the input Start_MT. Resetting Start_MT interrupts the motion task.
		1	Every edge at the input Start_MT starts a motion task. This allows to start a new motion task while another motion task is running without stopping the drive.
DBX83.1	Bit 1	0	With the input Accept_Value a parameter is read.
		1	With the input Accept_Value a parameter is written.
DBX83.2	Bit 2	0	Read and write of parameters is done with object numbers.
		1	Read and write of parameters is done with PNUs.