SIEMENS

SIMODRIVE 611 universal

Description **03.2004 Edition**

Function Block FB 83
for SIMODRIVE 611 universal ↔ S7-CPU

SIMODRIVE 611 universal

Description 03.2004 Edition

Function Block FB 83 SIMODRIVE 611 universal ↔ S7-CPU

Valid for
Function block FB 83, Version 04 and Step 7, SW 5.0

Contents

1. General Information

This block supports data transfer between a SIMODRIVE 611 universal and an S7-CPU via Profibus. The data interfaces are individually defined in a data block by having a structure with pre-configured UDTs.

The principal communications structure of a SIMODRIVE 611 universal with a Class 1 DP master is illustrated in **Fig. 1**. In this case, the process data (PZD) and the task data (PKW) are consistently transferred over the complete length of the PZD or PKW area. The control and return signals are defined in more detail in the User Interface Assignment.

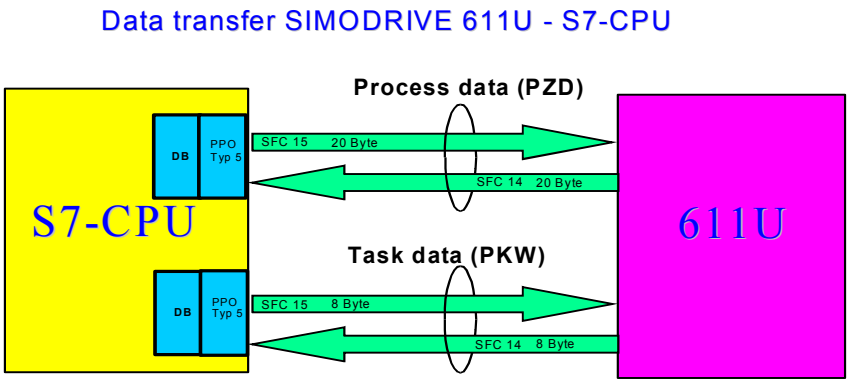


Fig. 1 Example: Consistent data transfer

PKW interface

The telegram structure of the PKW interface is illustrated in Fig. 2 (also refer to the Description of Functions, SIMODRIVE 611 universal).

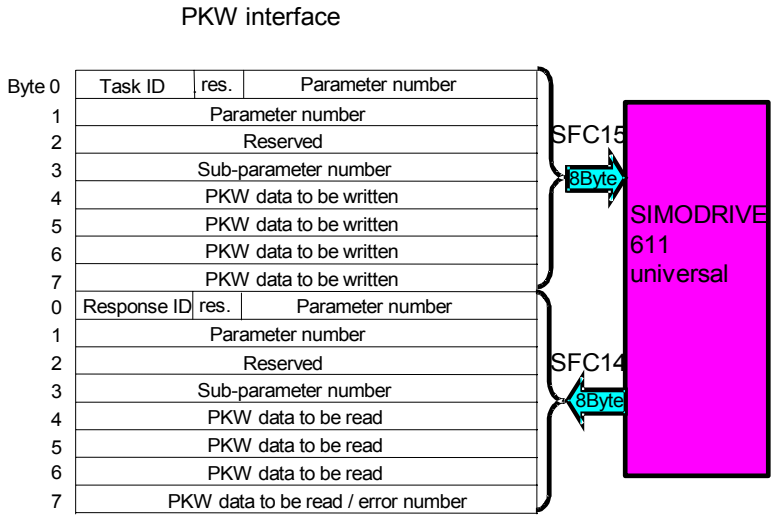


Fig. 2 Telegram structure

2. User Interface Structure

The principle structure of the user interface is illustrated in **Fig. 3**. In this case, an axis-specific data block with pre-configured UDTs must be generated for each axis. The process data area (UDT positioning operation or UDT nset) by which the SIMODRIVE 611universal is controlled or which mirrors its feedback (return) signal is mandatory. Optional data, for example, traversing blocks and fault messages can be individually integrated into the axis-specific data block depending on the particular requirement. FB83 must be cyclically called for each axis.

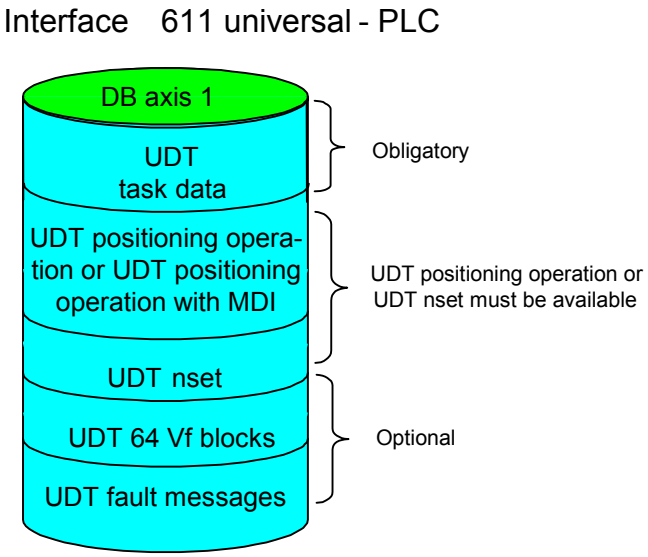


Fig. 3 Principle structure of the user interface

In the first S7-CPU cycle, the FB83 checks which data areas the user defined (**Fig. 4**). This is specified in the internal data. The S7-CPU must be re-started if the structure of an axis-specific DB is changed.

Notice

The FB_611 universal block must be called during the first PLC cycle. It this is not done, the UDT autoscan is not run, as this scan was realized with SFC6.

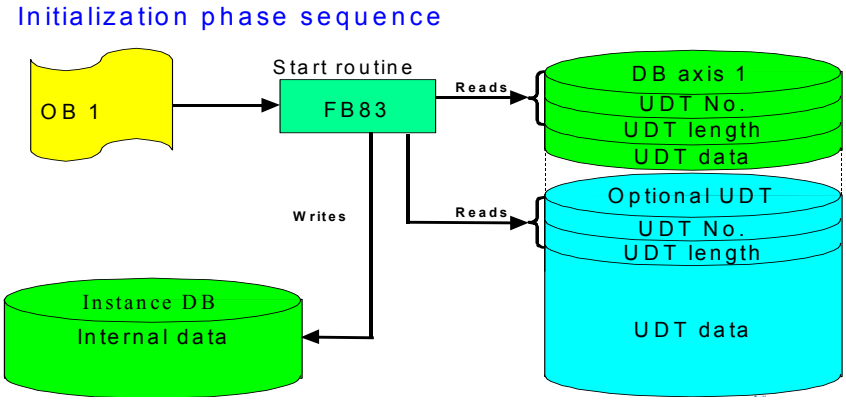


Fig. 4 Checking the data areas using FB 83

The structure of the obligatory UDT is illustrated in **Fig. 5**. The precise assignment of the UDT is provided in the User Interface Assignment.

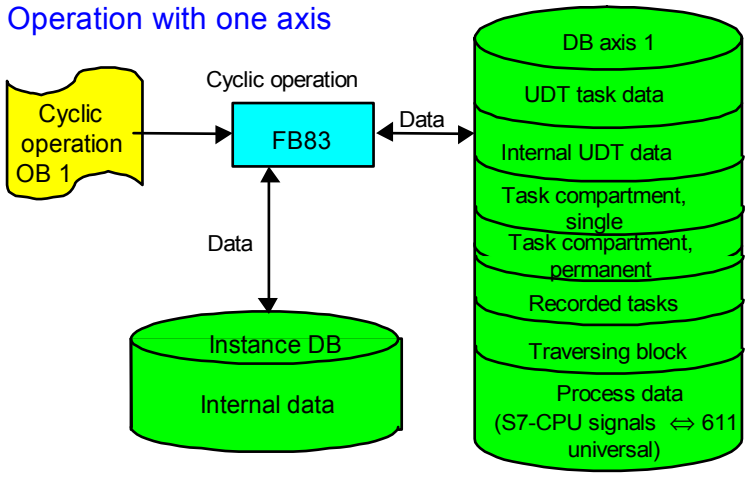


Fig. 5 Structure of the obligatory UDTs

For operation with n axes, only n+1 DBs are required if the internal data is saved in a multi-instance DB; refer to **Fig. 6.**

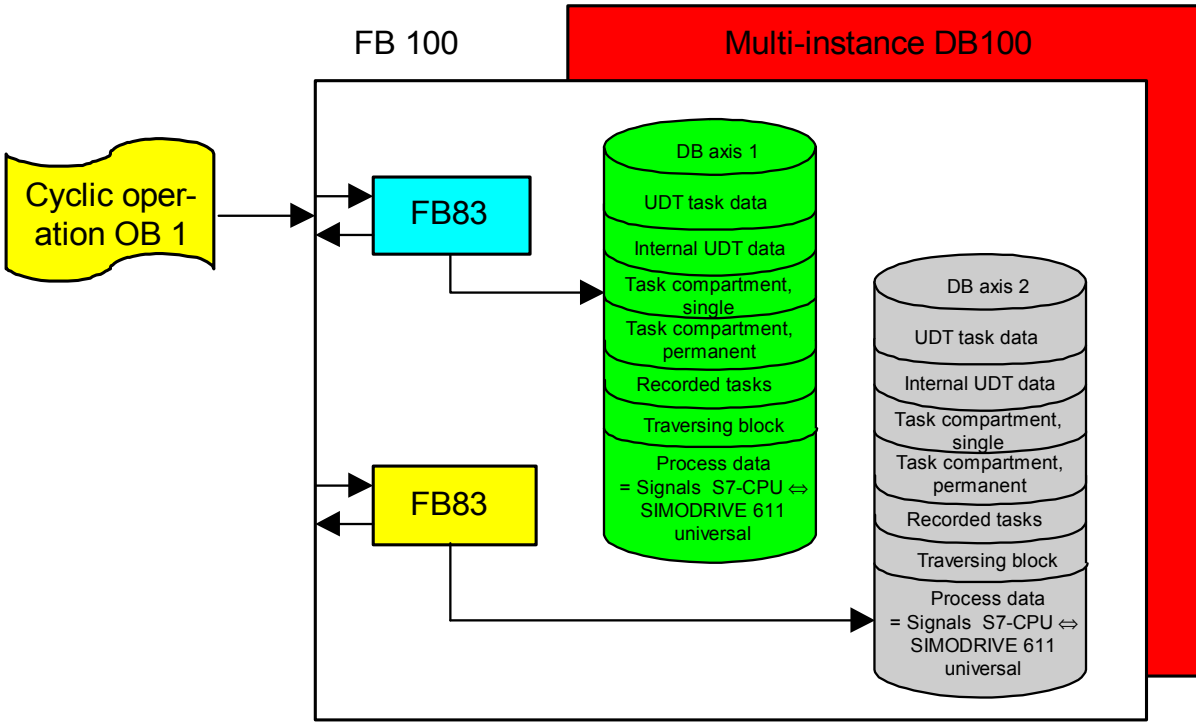


Fig. 6 Operation with n axes

3. Calling the Function Block

3.1. General information

The function block FB83 must be cyclically called for each axis. Communications is realized via the I/O address set in the hardware configuration.
In the configuration example (Section 7) when the FB is called the first time, it must transfer address 256 for axis A to the input variable LADDR and when called the second time, it must transfer address 284 for axis B. The number of the particular axis data block (axis DB No.) must be transferred to the FB.

Two Any Pointers must be transferred for each axis in order to transfer process data. The process data should be preferably in a DB (Any Pointer, for example: p#db71.dbx 166.0 byte 20). If the process data are to be saved in a DB, UDTs 30010 for positioning or UDT 30009 for closed-loop speed controlled operation can be used (this is recommended as the individual bits of the process data are symbolized in the UDT).

If the process data are to be saved in a DB, the following UDTs can be used:

- UDT 30010 for positioning operation
- UDT 30008 for positioning operation with MDI
- UDT 30009 for speed-controlled operation

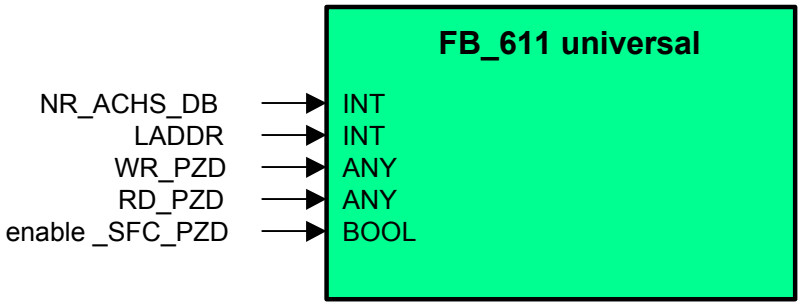


Fig. 7 Input parameters, SIMODRIVE 611 universal

Example of a call in STL for PPO type 5

```
FUNCTION_BLOCK FB 100
TITLE =
VERSION : 0.1


VAR
  a1 : FB 83;
END_VAR
BEGIN
NETWORK
TITLE =
CALL #a1 (
        NR_ACHS_DB        := 71,
        LADDR             := 256,
        WR_PZD            := P#DB71.DBX 166.0 BYTE 20,
        RD_PZD            := P#DB71.DBX 206.0 BYTE 20,
        enable_SFC_PZD    := TRUE);

 END_FUNCTION_BLOCK
```

Example of a call in STL for MDI (telegram type 110)

```
FUNCTION_BLOCK FB 100
TITLE =
VERSION : 0.1


VAR
  a1 : FB 83;
END_VAR
BEGIN
NETWORK
TITLE =
CALL #a1 (
        NR_ACHS_DB        := 71,
        LADDR             := 256,
        WR_PZD            := P#DB71.DBX 166.0 BYTE 24,
        RD_PZD            := P#DB71.DBX 206.0 BYTE 14,
        enable_SFC_PZD    := TRUE);

 END_FUNCTION_BLOCK
```

Conditions for setting the data consistency

The data consistency over the complete length is defined in parameter **enable_SFC_PZD**.

For a consistent data transfer over the word, the parameter set should be set to **false**. If consistent data transfer over the complete length is present, the parameter should be set to **true**.

The PKW data are always consistently transferred over the complete length independent of **enable_SFC_PZD**. Only the PZD area can be changed over. Refer to Configuring SIMODRIVE 611 universal in the S7 station when checking the data consistency.

Explanation of the formal parameters of FB_611 universal:

Signal	Type	Char.	Value range	Comment
NR_ACHS_DB	I	Int	Dependent on the CPU	Number of the data block for an axis DB
LADDR	I	Int		Start of the I/O address
WR_PZD	I	Any	P#Mm.n Byte x.. P#DBnr.dbxm.n Byte x	Target area for process data Master –> Slave (control words/setpoints) Valid for WR_PZD and RD_PZD: The point length depends on the PPO type. PPO No. of PZD bytes 1 4 2 12 5 20 If not changed, then PPO type 5 is set.
RD_PZD	I	Any	P#Mm.n Byte x.. P#DBnr.dbxm.n Byte x	Target area of process data, Master <– Slave (status words/actual values)
enable_SFC_PZD	I	Bool		True: The PZD area is "constant over the complete length" - process data is transferred in the area specified under WR_PZD/RD_PZD with SFC 14/15. False: The PZD area is consistent over the unit. Process data is transferred via load/transfer commands.

3.2. Configuration instructions

Initial situation: A new S7 project has been created. The hardware has been configured.

There are two procedures: Either you open project **611U_39** and change the project for the application, or you already have a functioning project and you only insert the interface.

Changing the 611U_39 project

1. Open the block folder and edit data blocks 71 and 72. Remove the UDTs which are not required. Remove axis data blocks which are not required (if only one axis is required).

2. Changes and expanded functionality can now be incorporated.

Inserting an interface in the project

1. Insert the FB 83 and the required UDTs into the existing S7 project.
   UDT30000 = basis UDT
   UDT30001 = edit 64 traversing blocks
   UDT30002 = read-out faults
   UDT30008 = positioning operation with MDI
   UDT30009 = closed-loop speed controlled operation
   UDT30010 = positioning operation

2. Create a data block for each axis which contains the required UDTs.
The UDT 30000 must always be included in the axis data block and be called at the first position.
Refer to the examples for positioning and closed-loop speed controlled operation.

Examples for the data block declaration:

Positioning operation

Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	al	UDT30000		Basis UDT
+156.0	p	UDT30010		Positioning operation UDT
+246.0	v	UDT30001		Traversing block UDT
+1538.0	s	UDT30002		Fault UDT
=2318.0		END_STRUCT		

MDI positioning operation

Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	a2	UDT30000		Basis UDT
+156.0	d	UDT30008		MDI positioning operation
+246.0	s	UDT30002		Fault UDT
=1026.0		END_STRUCT		

Closed-loop speed controlled operation

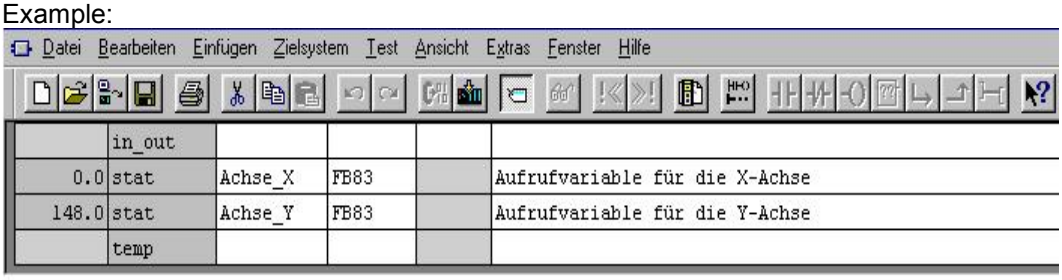
Address	Name	Type	Initial value	Comment
0.0		STRUCT		
+0.0	a2	UDT30000		Basis UDT
+156.0	d	UDT30009		Speed operation UDT
+246.0	s	UDT30002		Fault UDT
=1026.0		END_STRUCT		

You now have 2 possibilities:
A) Manage internal data using a multi-instance data block.
B) Assign each axis a dedicated data block.

A) Managing internal data with a multi-instance data block

Create a function block (call block).
For each axis, declare a call variable in the call block.
Declaration = stat, **Name** = freely selectable , **type** = FB 83

Example:



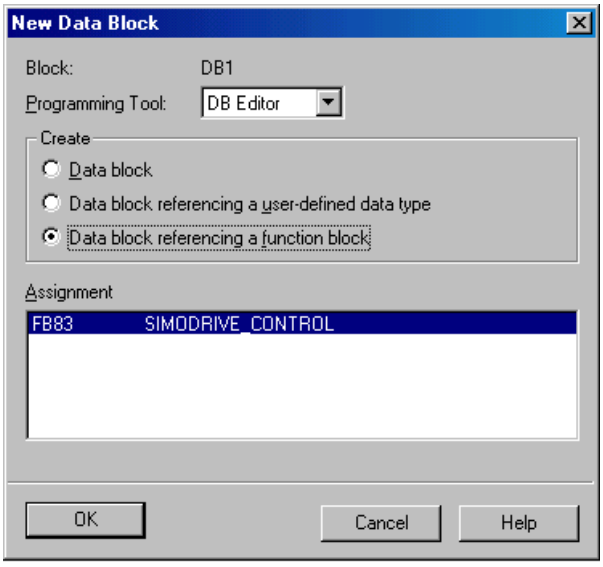
Call the interface FBs (FB 83) in this function block (call block).
FB 83 must be called once for each axis.
Set the call for the call block at the position it will be used.
Example: call FB 40, DB40 ➔ whereby DB40 is the new multi-instance DB of the interface

Set-up error OBs if required (OB 81, 82, 86, 87, 121, 122).

B) Assign each axis a dedicated data block

Call FB 83 in the program or in a new FC or FB.
Call command: call FB83 , DB XY → DB XY is the instance DB

Open the new data block and assign the FB83 as instance data block.



If changes are made at the data block with reference to the UDT arrangement, then the system must be restarted, otherwise autoscan does not register the changes.

Notice

Never change UDT headers or UDTs. If UDTs are changed, then read and write length errors can occur.

---

4. User Interface Assignment

4.1. List (axis DB)

Internal data								
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBW 0	UDT number							
DBW 2	UDT length							
DBW 4	Check number							
DBW 6	Reserve							
DBW 8	Reserve							
DBW 10	Reserve							

Error	
DBW 12	ErrorNumbr611U

Task-single PLC ⇔ 611U								
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBB 14	Error				Busy	Done	WR	RD
DBW 16	tasksi							
DBW 18	ind							
DBD 20	Data							
DBW 24	ErrorNumbr							
DBW 26	Reserve							
DBW 28	Reserve							
DBW 30	Reserve							
DBW 32	Reserve							

Tasks, permanent PLC ⇔ SIMODRIVE 611 universal								
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBB 34	Error				Busy	Done		RD
DBW 36	taskpe							
DBW 38	ind							
DBD 40	Data							
DBW 44	ErrorNumbr							
DBW 46	Reserve							
DBW 48	Reserve							
DBW 50	Reserve							
DBW 52	Reserve							

chain_task								
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBW 54	chain_PNU_1							
DBW 56	chain_IND_1							
DBD 58	chain_Data_1							
DBW 62	chain_PNU_2							
DBW 64	chain_IND_2							
DBD 66	chain_Data_2							
DBW 70	chain_PNU_3							
DBW 72	chain_IND_3							
DBD 74	chain_Data_3							
DBW 78	chain_PNU_4							
DBW 80	chain_IND_4							
DBD 82	chain_Data_4							
DBW 86	chain_PNU_5							
DBW 88	chain_IND_5							
DBD 90	chain_Data_5							
DBW 94	chain_PNU_6							
DBW 96	chain_IND_6							
DBD 98	chain_Data_6							
DBW 102	chain_PNU_7							
DBW 104	chain_IND_7							
DBD 106	chain_Data_7							
DBW 110	chain_PNU_8							
DBW 112	chain_IND_8							
DBD 114	chain_Data_8							
DBW 118	chain_PNU_9							
DBW 120	chain_IND_9							
DBD 122	chain_Data_9							
DBW 126	chain_PNU_10							
DBW 128	chain_IND_10							
DBD 130	chain_Data_10							

Complete traversing block								
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DBB134	Mode	Command parameter	Command	Deceleration override	Acceleration override	Velocity	Position	Block number
DBW 136	Block number							
DBD 138	Position							
DBD 142	Velocity							
DBW 146	Acceleration override							
DBW 148	Deceleration override							
DBW 150	Command							
DBW 152	Command parameter							
DBW 154	Mode							

4.2. Description

Address	Symbol. Name	Description
DBW 0	UDT Nummer	Number of the UDTs from here. Initial value: 30000
DBW 2	UDT Länge	Length of the UDTs from here in bytes. Initial value: 156
DBW 4	Kontrollnummer	Initial value: 30000
DBW 12	ErrorNumbr611U	Error number of 611 univ. is saved here.
DBX 14.0	RD	Reading data
DBX 14.1	WR	Writing data
DBX 14.2	Done	Feedback, task completed
DBX14.3	Busy	Task is being processed.
DBX 14.7	Error	Task cancelled with error.
DBW 16	tasksi	Task number, this can be a parameter number or special task number.
DBW 18	ind	Sub-parameter number. Refer to the Description of Functions, SIMODRIVE 611 universal, Section, Parameter lists.
DBD 20	Data	Value which is to be written or value read-out after a read task.
DBW 24	ErrorNumbr	Error number of an error which occurred when executing the task. -1 means no error. Refer to Section 8
DBX 34.0	RD	Reading parameter.
DBX 34.2	Done	Feedback, task completed
DBX 34.3	Busy	Task is being processed.
DBX 34.7	Error	Task cancelled with error.
DBW 36	taskpe	Special task number
DBW 38	Ind	Sub-parameter number. Refer to the Description of Functions, SIMODRIVE 611 universal, Section Parameter lists.
DBD 40	Data	Reserved
DBW 44	ErrorNumbr	Error number for task execution. –1 means no error.
DBW 54, 62 ..., 126	chain_PNU_1-10	Parameter number for permanently reading-out a piece of data
DBW 56, 64 ..., 128	chain_IND_1-10	Sub-parameter number associated to the parameter number.
DBD 58, 66, ..., 130	chain_Data_1-10	Parameter value transferred from the 611 univ.
DBX 134.0	Satznum	Preselection, which parameters are to be transferred (134.1 to 134.7)
DBX 134.1	pos	Select, position
DBX 134.2	gesch	Select, velocity
DBX 134.3	beschover	Select, acceleration
DBX 134.4	verzover	Select, deceleration
DBX 134.5	bef	Select, command
DBX 134.6	befpara	Select, command parameter
DBX 134.7	mod	Select, mode
DBW 136	Satznummer	Value, block number
DBD 138	Position	Value, position
DBD 142	Geschw	Value, velocity
DBW 146	Beschl_over	Value, acceleration override
DBW 148	Verzoeg_over	Value, deceleration override
DBW 150	Befehl	Value, command
DBW 152	Befehlsparameter	Value, command parameter
DBW 154	Modus	Value, mode

5. Traversing Task Processing

Various tasks can be started using the interface block:

- Individually read/write into parameters
- Read-out fault memory
- Read/write individual traversing blocks
- Read/write traversing blocks
- Pre-assign traversing blocks 0..63
- Permanently read-out data

Important: The task may only be initiated if all of the task data are available.

Single task interface

Using the single task interface, tasks can be initiated which are only executed once. The status bit is set to busy while the task is running. If the task has been completed, busy is reset and the Done bit is set.

If an incorrect parameter number (tasksi), an incorrect sub-parameter number (IND) or an incorrect piece of data was entered, the error bit is set and in the error compartment, an error number is specified which is described in the Description of Functions SIMODRIVE 611 universal in Sections 5, 7 and 8.

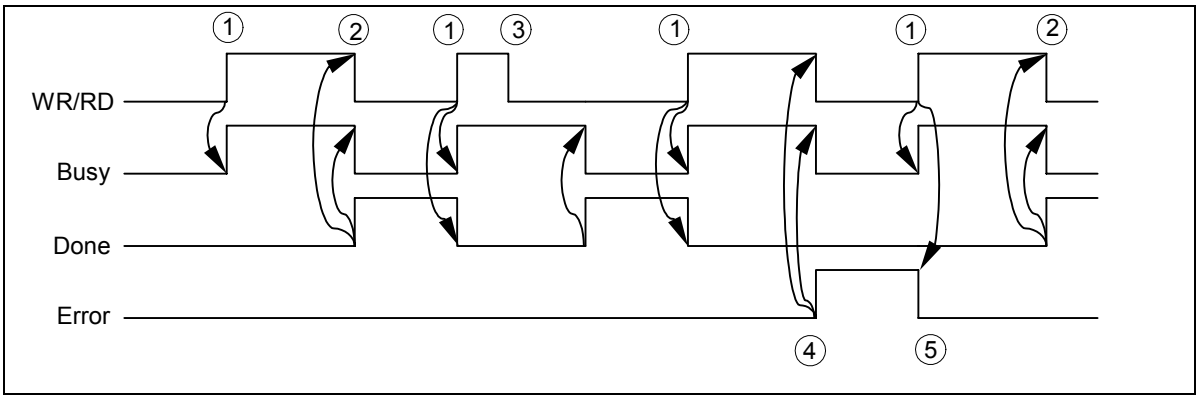


Fig. 8 Structure of the obligatory UDT

① Read or write tasks started by the user
② WR/RD request from FB611 universal automatically reset
③ A task is completed even if the request is withdrawn
④ The request is reset for a data transfer error
⑤ A new task resets the error message

Permanent task interface

Using the permanent task interface, tasks are initiated which are continually read-out (e.g. position actual value during axis motion). If an incorrect parameter number (taskpe), or an incorrect sub-parameter number (IND) was entered, the error bit is set and in the error compartment, an error number is specified which is described in the Description of Functions SIMODRIVE 611 universal in Sections 5, 7 and 8.

Task management

Using the 611 universal task interface **single**, two types of tasks can be started:

- All of the parameters from SIMODRIVE 611 universal
  (refer to the parameter list in the attachment to the Description of Functions SIMODRIVE 611 universal, Sections 5, 7 and 8)
- Special tasks

Reading individual parameters

Pre-conditions:	singl.tasksi = parameter number singl.Ind = sub-parameter number
Initiate:	single.RD = with 1
End:	singl.Done = TRUE
Output:	singl.Data = parameter value which is read-out

Write individual parameters

Pre-conditions:	singl.tasksi = parameter number singl.Ind = sub-parameter number singl.Data = value to be written
Initiate:	singl.WR = with 1
End:	singl.Done = TRUE

Read-out fault memory

Pre-conditions:	tasksi = 30002 singl.Ind = is ignored
Initiate:	singl.RD = start reading-out with 1
End:	singl.Done = TRUE
Values:	Fault cases and the number of faults are saved in the associated data block in UDT 30002.

Read/write individual traversing blocks

Pre-conditions:	tasksi = 30000 singl.Ind = select traversing block number (0 to 63) Bit bar 134.0 to 134.7 = pre-selection as to which parameters are to be transferred
Initiate:	singl.RD = read with 1 singl.WR = write with 1
End:	singl.Done = TRUE

Read/write traversing blocks

Pre-conditions:	tasksi = 30001 singl.Ind = first traversing block number singl.Data = last traversing block number
Initiate:	singl.RD = read with 1 singl.WR = write with 1
End:	singl.Done = TRUE
Values:	Read = values are saved in the assoc. data block UDT 30001 Write = values are transferred from the data block to the SIMODRIVE 611 universal

---

Notice

All data are only saved in the SIMODRIVE 611 universal user memory.

This is why the data must be saved in the FEprom via the parameter P 0652. Otherwise, the data are deleted when the system is switched on again.

Save data: Parameter P 0652 = 1

Pre-assign traversing blocks 0...63

Pre-conditions:	tasksi = 30011 singl.Ind = is ignored The block numbers are pre-assigned 0 to 63
Initiate:	singl.WR = write with 1
End:	singl.Done = TRUE

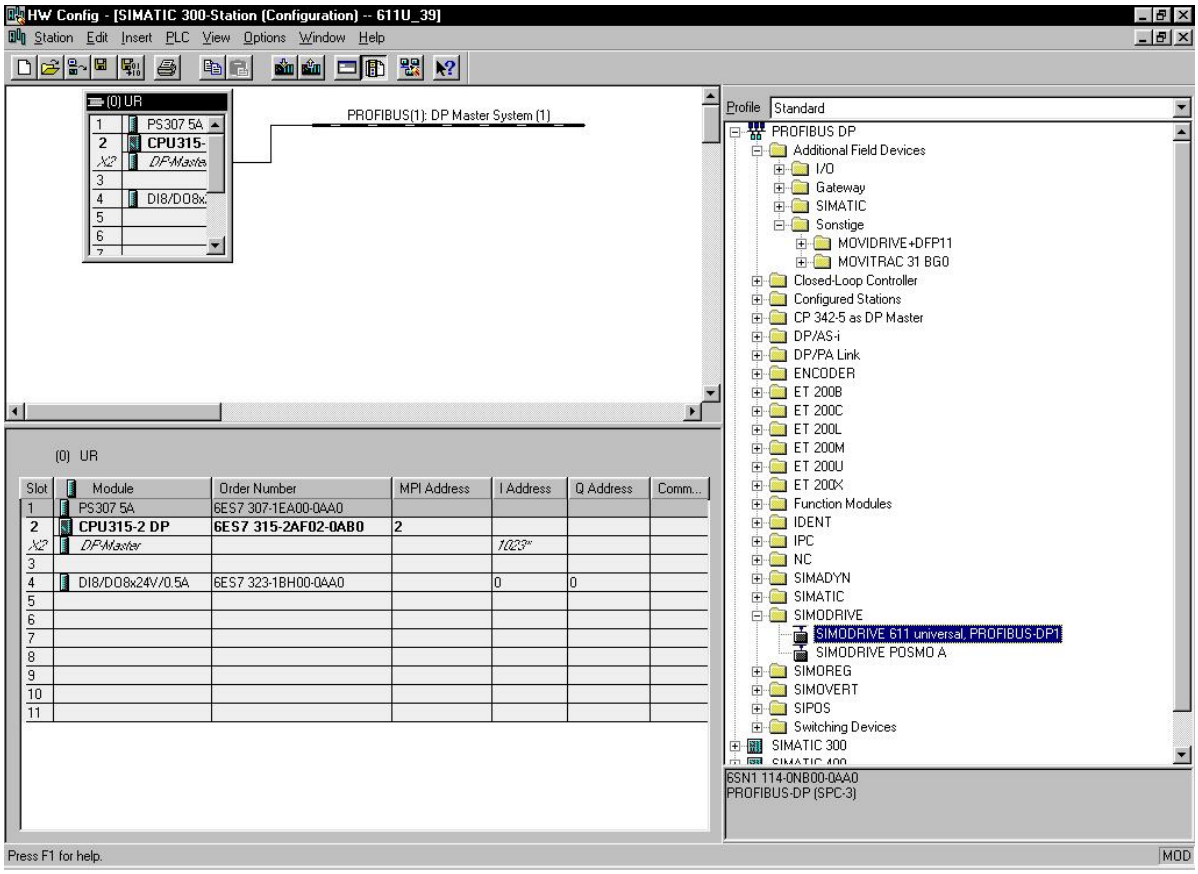
Permanently read-out data

Pre-conditions:	perm.taskpe = 31000 perm.Ind = is ignored chain_PNU_xy = parameter number of the value to be read-out chain_Ind_xy = sub-parameter number
Initiate:	perm.RD = with 1, permanently read the data as long as a 1 is present.
End:	Not relevant as the data are permanently read-out.
Values:	chain_Data_xy Values are in the particular data compartments of the parameter numbers.

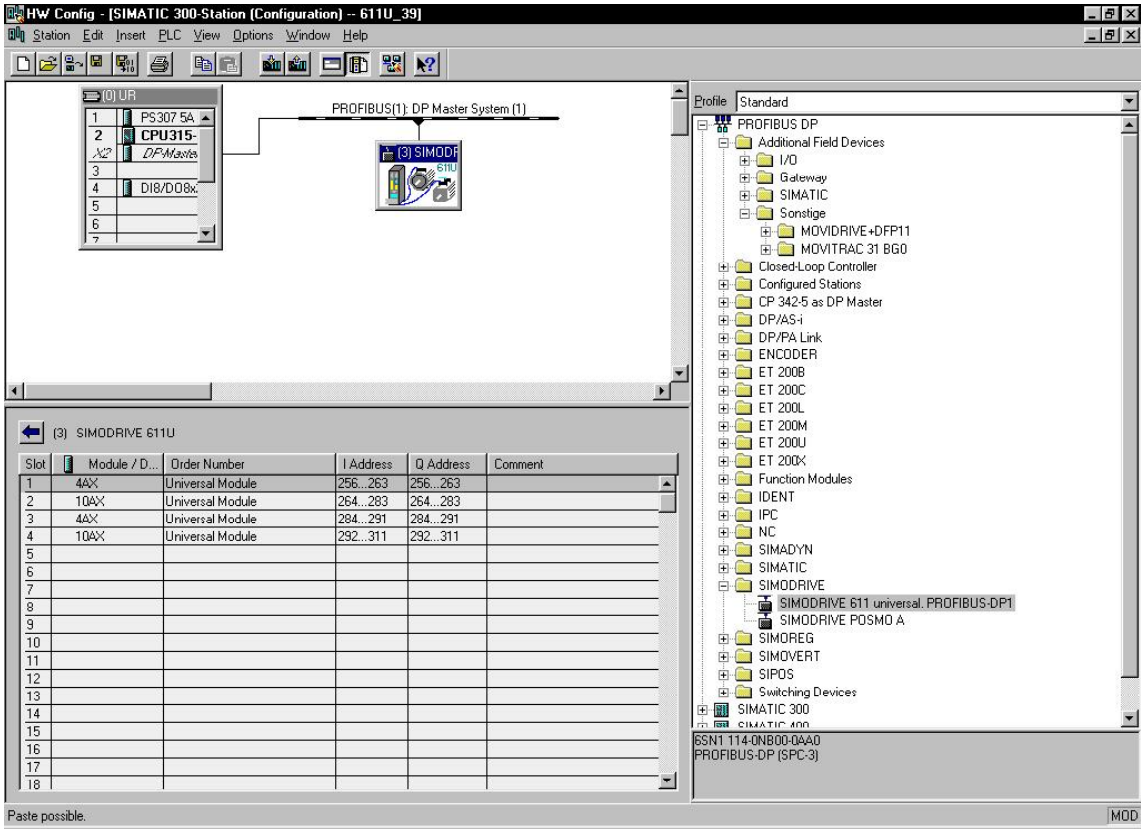
6. Configuring

6.1. PPO type 5

After configuring the S7 (for example a 315-2DP) the following is displayed in the hardware configuration:



Insert the SIMODRIVE 611 universal module at PROFIBUS-DP.
Then select the number of axes and the required PPO type.



When selecting a second 2-axis module with PPO type 5, your hardware configuration looks like this. The I/O addresses 256 to 283 are assigned to drive A. I/O addresses 284 to 311 are associated with drive B. Addresses 256 to 263 and 284 to 291 are the PKW (parameter ID value) data (task data telegram, 4-words long). The PZD (process data, 10-words long) are transferred to Profibus at I/O addresses 264 to 283 and 292 to 311.

CAUTION!
The address areas may not be separated. PKW and PZD must merge seamlessly with one another without any gaps.

6.2. Telegram type 110 – block positioning and MDI

Prerequisites:
SIMODRIVE 611 universal, Firmware Version V7.1 and parameterization of telegram type 110 with Simocom U.

The structure of telegram type 110 and handling of the MDI mode is described in the Description of Functions SIMODRIVE 611 universal, Sections 5.2 and 6.2.12.

To configure the telegram type 110, the GSD file si02808f.gsd or siem808f.gsd included in the toolbox 7.01 must be integrated in the hardware configurator: Options menu.

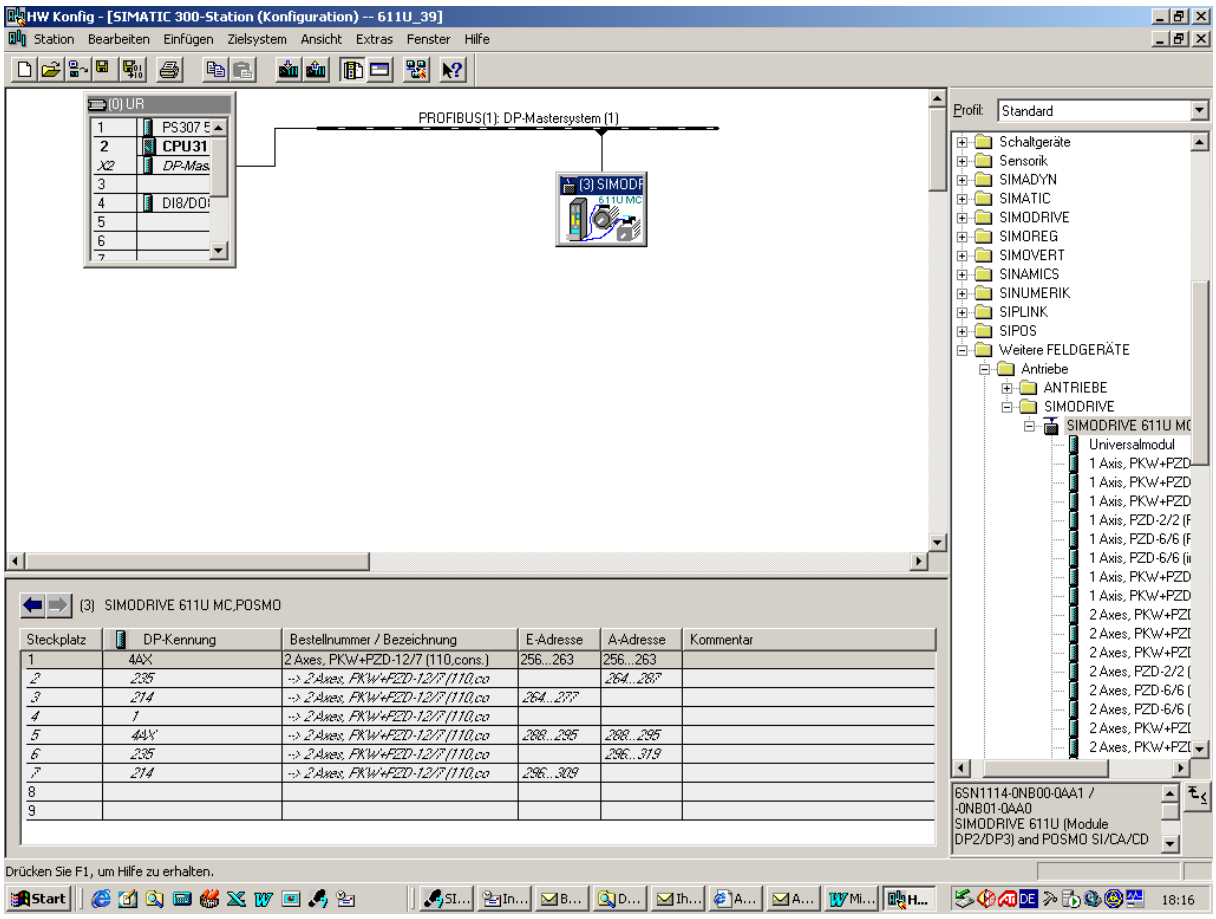
The appropriate assignment is then selected via the hardware catalog.

Notice

FB83 can only function with identidal initial addresses for the I/O area.

When selecting a double-axis module, differing initial addresses are automatically allocated for the I/O area for the second axis since these have a different length with telegram type 110.

The initial addresses must therefore be set manually.



7. Error when Executing a Task

If an error is specified when executing a task (**singl.Error** = true), then an evaluation can be made using the **singl.ErrorNumbr** parameter. The significance of the error codes is described in the Description of Functions, SIMODRIVE 611 universal, Section 5.

Error ID	Error cause
0	Illegal parameter number (the parameter does not exist)
1	Parameter value cannot be changed (parameters can only be read or are write protected)
2	Lower or upper value limit violated
3	Erroneous sub-index
4	No array (parameter does not have a sub-parameter)
5	Incorrect data type (this is not required for type conversion)
6 to 19	Not required
20 to 100	Reserved

8. Testing Aids

The following variable tables must be used to activate the MDI function of SIMATIC:
VAT_MDIAchseA / VAT_MDIAchseB

The travel data blocks are changed via the following variable tables:
VAT_Posit_AchseA / VAT_Posit_AchseB

To
SIEMENS AG

A&D MC BMS
Postfach 3180

D-91050 Erlangen

Suggestions

Corrections

for document:

SIMODRIVE 611 universal

Function Block FB 83
for SIMODRIVE 611 universal
↔ S7-CPU

From

Name

Company address/Dept.

Street

Postal code: City:

Telephone: /

Telefax: /

Description

Edition: 03.2004

Should you come across any printing errors when reading this publication, please notify us on this sheet.

Suggestions for improvement are also welcome.