Tool for Control Technology

applications
& TOOLS

Tools Collection of Functions for Programming Tasks

SIEMENS

Tools Collection of Functional Examples for Date and Time

Note

The application examples and Tools are not binding and do not claim to be complete regarding the circuits shown, equipping and any eventuality. The application examples and Tools do not represent customer-specific solutions. They are only intended to provide support for typical applications. You are responsible in ensuring that the described products are correctly used. These application examples and Tools do not relieve you of the responsibility in safely and professionally using, installing, operating and servicing equipment. When using these application examples, you recognize that we cannot be made liable for any damage/claims beyond the liability clause described. We reserve the right to make changes to these application examples at any time without prior notice. If there are any deviations between the recommendations provided in these application examples and other Siemens publications - e.g. Catalogs - then the contents of the other documents have priority.

Warranty, Liability and Support

We do not accept any liability for the information contained in this document.

Any claims against us - based on whatever legal reason - resulting from the use of the examples, information, programs, engineering and performance data etc., described in this application example shall be excluded. Such an exclusion shall not apply in the case of mandatory liability, e.g. under the German Product Liability Act (“Produkthaftungsgesetz”), in case of intent, gross negligence, or injury of life, body or health, guarantee for the quality of a product, fraudulent concealment of a deficiency or breach of a condition which goes to the root of the contract (“wesentliche Vertragspflichten”). However, claims arising from a breach of a condition which goes to the root of the contract shall be limited to the foreseeable damage which is intrinsic to the contract, unless caused by intent or gross negligence or based on mandatory liability for injury of life, body or health The above provisions does not imply a change in the burden of proof to your detriment.

For questions about this document please use the following e-mail address:

online-support.automation@siemens.com

Preface

In this example we introduce fully functional and tested automation configurations based on Siemens Industry Sector standard products and individual function blocks or tools, for simple, fast and inexpensive implementation of automation tasks.

Apart from a list of all required hardware and software components and a description of the way they are connected to each other, the examples include the tested tools or function blocks. This ensures that the functionalities described here can be reset in a short period of time and thus also be used as a basis for individual expansions.

Industry Automation and Drives Technologies Service & Support Portal

This entry is from the internet service portal of Siemens AG, Industry Automation and Drives Technologies. Clicking the link below directly displays the download page of this document.

http://support.automation.siemens.com/WW/view/en/31696259

Table of Contents

1 Time

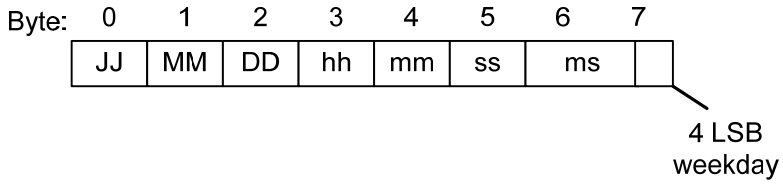
1.1 Determining the calendar day

Description

The calendar day is the number of past days from January 1st of a specific year to a specific date of the same year. The “DATE” data type can be used to calculate the calendar day. The “DATE” type contains the number of past days since 01.01.1990 and has a size of 16 bits. This allows to specify a date until max. 31.12.2168. When correcting the value by –1, the difference between a specific date and January 1st of the same year results in the number of days until this date. Leap years and the exception rules have already been considered internally in DATE format so that case differentiations are not necessary.

January 1st of a year can be determined with the “Date_and_Time” (DT) data type. Figure 1-1 shows the structure of the DT data type. Bytes 1 and 2 contain the month and day that are manually set to “1”. The necessary format conversions between “DT” and “DATE” can be performed with the “D_TOD_DT” (FC 3) and “DT_DATE” (FC 6) IEC functions.

Figure 1-1



“CldrDay” function

The “CldrDay” (FC1) function determines the corresponding calendar day of a transferred date in “DATE” format. The date is transferred with the “ddDate” input parameter. In this case, the “bSystemClock” input parameter must be “FALSE”. If “bSystemClock” is “TRUE”, the date is ignored at the “ddDate” input and the function determines the calendar day from the date that corresponds to the CPU system time. The respective calendar day is returned via the “iCalenderDay” output parameter. It is of the INT type. The function checks the date for permissible values and, in the event of an error, returns the value W#16#8001 via the “RET_VAL” return value.

Note Due to the internal conversion to DT format, the latest permissible date is 31.12.2089, even if the “DATE” data type allows a later date.

To determine the system time, the “CldrDay” function uses the “READ_CLK” (SFC1) system function. If the return value of this function is not equal to 0, the function is terminated and the error code is returned via RET_VAL.

Block parameters of the “ CldrDay ” (FC 1) function

Table 1-1

Parameter	Declaration	Data type	Area	Description
bSystemClock	IN	BOOL	I, Q, M, D, L	true: Function determines the calendar day from the current date of the system time false: Function determines the calendar day of the date at “ddDate”.
ddDate	IN	DATE	I, Q, M, D, L	The calendar day is calculated from this date if “bSystemClock” == true. If “bSystemClock” == false, this parameter has to significance
iCalenderDays	OUT	INT	Q, M, D, L	Calendar days that passed in the relevant year
RET_VAL	Return	INT	Q, M, D, L	Error code: W#16#8001: Value at “ddDate” is not within the permissible limits. For other error messages, see documentation of the SFC1 system function

Example

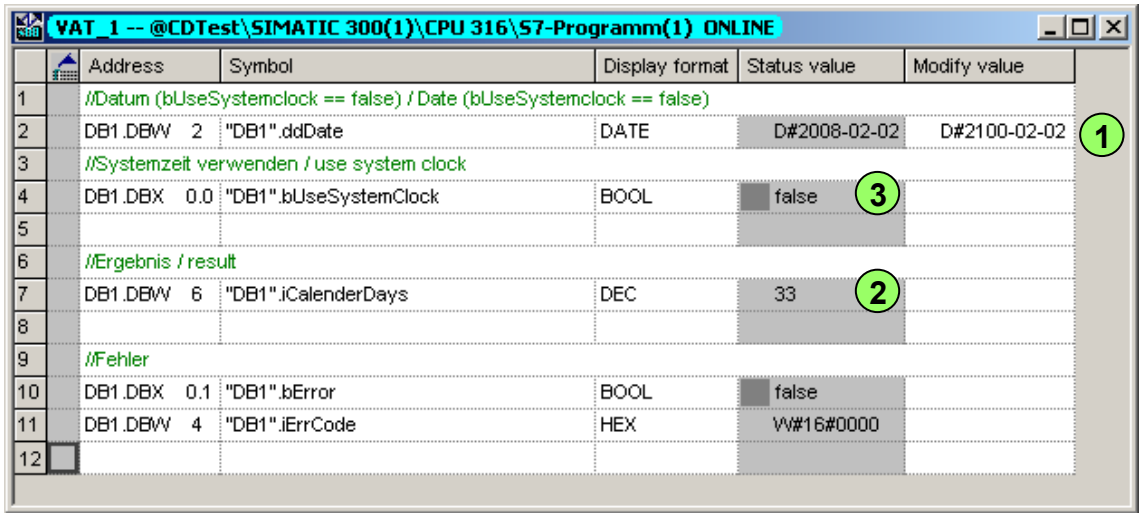
In the program example, the “CldrDay” (FC1) function is called in “TestFB” (FB1) with all necessary parameters and an error check is performed. If an error occurs, this is indicated via a bit. The input parameters can be changed and the results can be checked in the “VAT_1” variable table created in the project.

To test the sample project, proceed as follows:

Table 1-2

Step	Action/event
1.	✓ Download the complete station to the CPU or to S7-PLCSIM and open the "VAT_1" variable table in online mode
2.	✓ In the variable table, enter a valid date in "DB1".ddDate (1) **Result:** The number of calendar days is displayed in "DB1".iCalenderDays. In the example, these are 33 days for 02.02.2008. (2)
3.	✓ Set the "DB1".bUseSystemClock bit. (3) **Result:** The calendar day of the current system time is determined instead of the transferred date.

Figure 1-2



Technical specifications

Table 1-3

Block	Data
CldrDay (FC1) determination of the calendar day	Required local data : 30 bytes Load memory requirement : 448 bytes Work memory requirement : 356 bytes

1.2 24-hour time switch

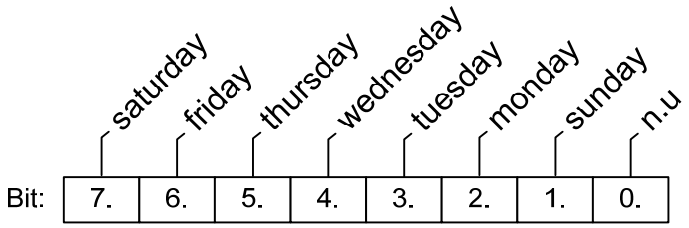
Description

For time-dependent switching operations, a simple time switch with a maximum time range of 24 hours is sufficient in many cases. By specifying the switch-on and switch-off time and the weekdays, such a time switch can be used flexibly, also beyond the day limits (e.g., from 23:00 to 01:00 of the following day). The possibility of activating the switching times for individual weekdays increases flexibility.

"TSwitch" (FC 100) function

The "TSwitch" function is the realization of a 24-hour time switch. By specifying switch-on time ("OnTime"), switch-off time ("OffTime") and days ("SwitchDays"), you can parameterize the switching time. The weekdays on which the time switch is active are specified as a bit pattern:

Figure 1-3



For example, the value B#16#C (bit 2 and 3 set) means that switching is performed only on Mondays and Tuesdays.

The function is processed only if the "Enable" input == "true". If the switch-off time is less than the switch-on time, this means that switching is performed beyond the day limit.

For example,

- SwitchDays = B#16#80 (only Saturday),
- TimeOn = TOD#23:00:00.000,
- TimeOff = TOD#01:00:00.000

mean that the time switch is switched on on Saturday at 23:00 and switched off on Sunday at 1:00. If switching is to be performed on Saturdays only, bit 1 (= Sunday) must not be set in this case since otherwise the time switch would also be switched on on Sunday at 23:00 and switched off on Monday at 01:00. The "WeekDayToPattern" (FC101) auxiliary function features a binary input for each weekday and provides a bit pattern for the weekdays corresponding to the inputs at the output.

Switching output "Q" is enabled as soon as "TimeOn" <= time <= "TimeOff". "Q" is available as an inverted "QN" output signal.

Note

- If several switching times are required, the function can be called multiple times.
- The function internally calls the SFC1 (“READ_CLK”) system function and the FC8 (“TD_TOD”) IEC function.

Block parameters of the “TSwitch” function

Table 1-4

Parameter	Declaration	Data type	Area	Description
OnTime	IN	TOD	I, Q, M, D, L	Switch-on time
OffTime	IN	TOD	I, Q, M, D, L	Switch-off time
SwitchDays	IN	BYTE	I, Q, M, D, L	Weekdays on which switch-on and switch-off time 1 are evaluated
Enable	IN	BOOL	I, Q, M, D, L	Enable block (“1”), disable block (“0”). If the block is disabled, W#16#8001 is returned as a return value.
Q	OUT	BOOL	Q, M, D, L	Switching output “1” if switch-on condition is met
QN	OUT	BOOL	Q, M, D, L	Inverted switching output “0” if switch-on condition is met
RET_VAL	RET_VAL	INT	Q, M, D, L	Error code: W#16#0 : No error W#16#8001 : Block disabled <>0 : Error for SFC1

Example

In the program example, the “TSwitch” (FC100) function is called in “TestFB” (FB1) with all necessary parameters and an error check is performed. If an error occurs, this is indicated via a bit. The input parameters can be changed and the results can be checked in the “VAT_1” variable table created in the project.

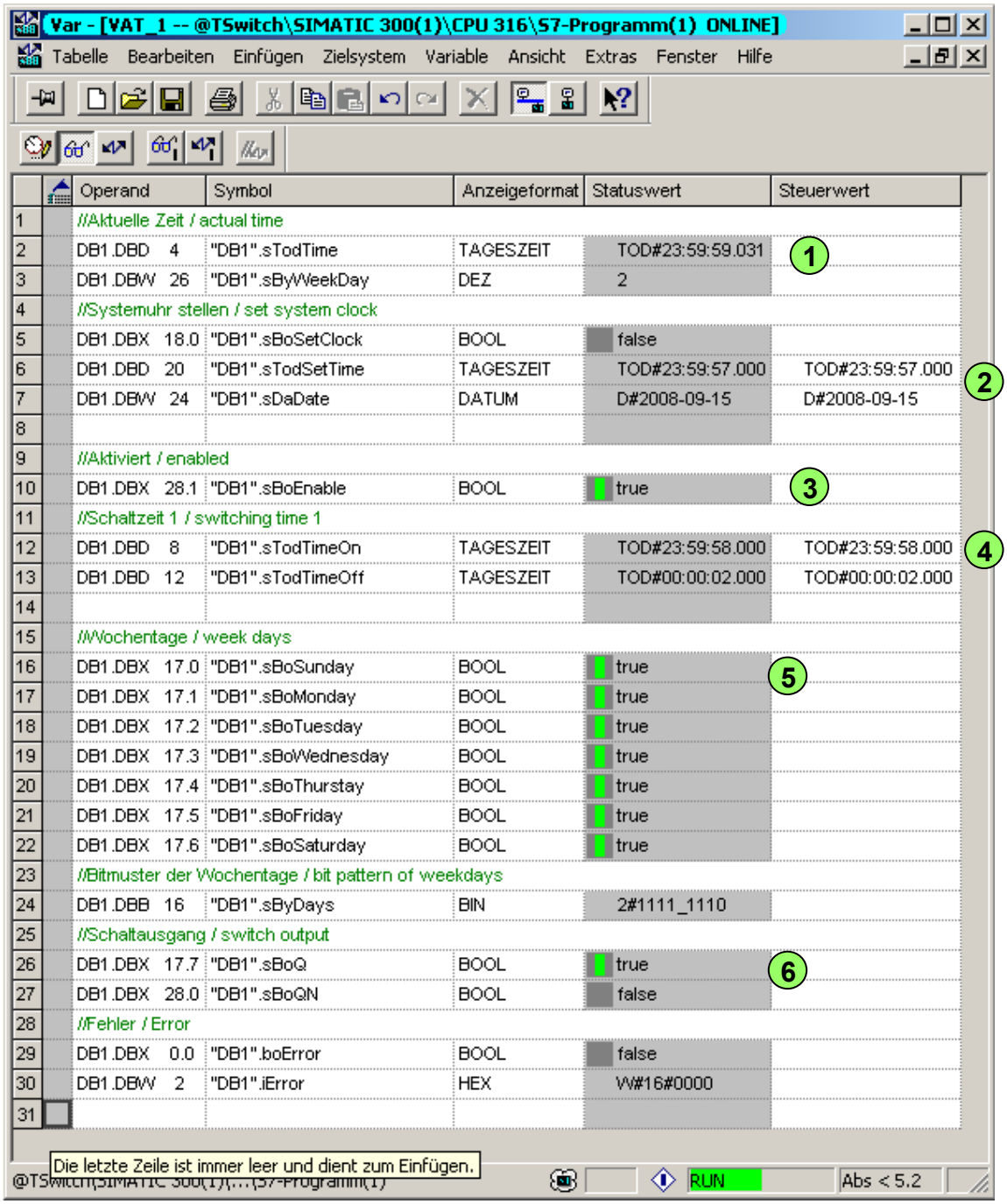
NOTICE **For test purposes, the following sample project provides the option of resetting the CPU system time. For this reason, the following steps must only be performed with PLCSIM.**

To test the sample project, proceed as follows:

Table 1-5

Step	Action/event
1.	✓ Download the complete station to S7-PLCSIM and open the “VAT_1” variable table in online mode
2.	✓ Read the current system time and the weekday 1 ✓ If you want to use another time for test purposes, enter the desired time and the desired date, enable the “sBoSetClock” input and disable it. 2 Result: The system time is set to the specified value 1
3.	✓ Enter the desired switching time 4 ✓ Select the weekdays on which the switch-on and switch-off time are evaluated. 5
4.	✓ Activate the function 3
5.	✓ Monitor the switching output 6 Result: As soon as the switching time is reached, the “Q” output changes from “0” to “1”

Figure 1-4



Technical specifications

Table 1-6

Block	Data
TSwitch (FC100) 24-hour time switch	Required local data : 30 bytes Load memory requirement : 534 bytes Work memory requirement : 418 bytes

For the associated download file, please refer to the chapter “Overview of the Download Files”.

1.3 Time-of-day schedule

Description

If more complex switching operations are to be realized with a time switch, it is mostly not sufficient to use a 24-hour time switch with only one single switching point as described in 1.2. Amongst others, the use of the time switch described in the following allows the following switching points:

- Every hour (at a specific minute)
- Every day or only on a specific weekday
- Every month or only at a specific month
- Only on weekdays or only on weekends

When combining these options, for example, the following switching times are possible

- Only on weekends at 8:00 h for each month
- On each first of the month every full hour
- In January from Monday to Friday at 12:00 h
- Always on Friday at 18:25 h

Several switching points of this time switch can be managed in a schedule.

"TOD" (FC1) function

The "TOD" function is the realization of a time switch that can manage up to 255 switching points in a schedule in a data block. The switching points are created as a structure (UDT). The structure is described in the next section. The block is created as an FC and has the following formal parameters:

- DBNO: Number of the data block containing the schedule with the switching points.
- ENT: Number of switching points included in the DNBO data block.
- ERROR: Bit that is set when an error occurs
- CODE: Detailed message when an error occurs (0 when no error occurs)

Note

- The function should be called at least once per minute. It may be continuously processed in OB 1 “CYCL_EXC”, scheduled for execution in OB 10 to OB 17 “TOD_INTx” on a “per-minute” basis, or called from OB 30 to OB 38 “CYC_INT*y*” on a periodic basis.
- An event is active only while this switch-on condition is met. To be able to specify, for example, an exact switch-on time and an associated switch-off time, separate entries have to be created for the switch-on and switch-off time and the positive edges of the “EVENT.active” events (see structure of the UDT1 schedule entries) must be evaluated.

Structure of the “SCHED” (UDT1) schedule entries

The schedule structure for an entry is defined in the “SCHED” UDT and is as follows:

Table 1-7

```
ATD                 STRUCT      //structure for activation
time/date
      minute        INT         //scheduled minute
      hour          INT         //scheduled hour
      day           INT         //scheduled day of the month
      month         INT         //scheduled month
      dow           INT         //scheduled weekday
                    END_STRUCT

EVENT               STRUCT
      id            DINT        //event identifier assigned by
user
      active        BOOL        //current event entry is active
                    END_STRUCT
```

ATD and EVENT are structures and ATD (activation time and date) contains the scheduled activation time and date information. EVENT contains information on the event such as activation and a user-assigned identifier.

The significance of the entries is described below:

Table 1-8

Entry	Significance
ATD.minute	This entry defines the minute (0..59) at which the event is to be active. “-1” marks every minute as valid. Other values are not permitted.
ATD.hour	This entry defines the hour (0..23) at which the event is to be active. “-1” marks every hour as valid. Other values are not permitted.
ATD.day	This entry defines the calendar day of a month (1..31) on which the event is to be active. “-1” marks every calendar day as valid.

Entry	Significance
	Other values are not permitted.
ATD.month	This entry defines the month (1 = January – 12 = December) at which the event is to be active. “-1” marks every month day as valid. Other values are not permitted.
ATD.dow	This entry defines the weekday (1 = Sunday – 7 = Saturday) on which the event is to be active. “-1” marks every weekday as valid. Other values are not permitted. If the event is to be active only from Monday to Friday, the value “8” has to be entered. If the event is to be active only on weekends (Saturday/Sunday), a value of “9” has to be entered Other values are not permitted.
EVENT.id	This entry assigns a unique identifier to the event. The value is assigned by the user (optional) and does not affect the function.
EVENT.active	This entry is set by the “TOD” function (true) when the event is active. The entry is “false” when the event is not active.

For a sample configuration of a data block with two entries, the result for the first “Data”.entry[1]” with the values

- ADT.minute = 30
- ADT.hour = -1
- ADT.day = -1
- ADT.month = -1
- ADT.dow = -1

is a switching instant for each hour on each day in each month when the value of the minute is 30.

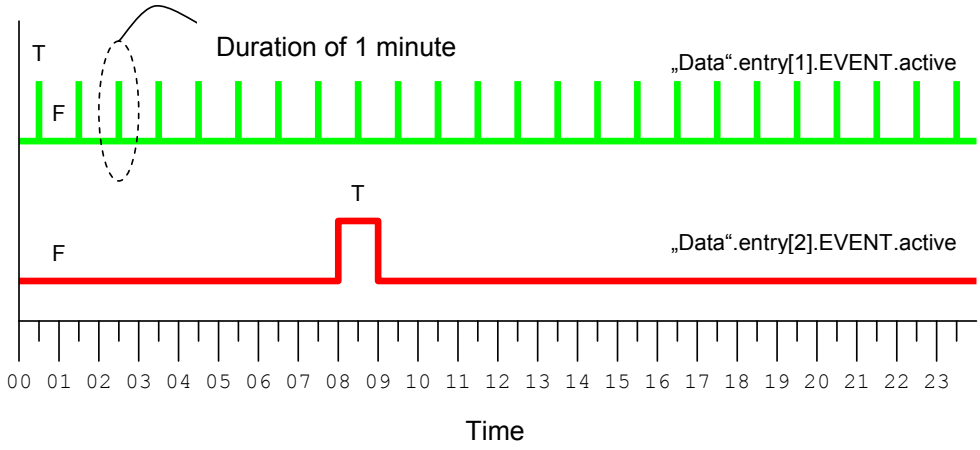
For a sample configuration for the second “Data”.entry[2]” with the values

- ADT.minute = -1
- ADT.hour = 8
- ADT.day = -1
- ADT.month = -1
- ADT.dow = 8,

the “Data”.entry[2].EVENT.active” event is activated on each weekday in each month at 8:00 h for one hour.

The figure below shows when and how long the events are active for this sample configuration.

Figure 1-5



Block parameters of the “ TOD ” function

Table 1-9

Parameter	Declaration	Data type	Area	Description
DBNO	IN	INT	I, Q, M, D, L, const.	Number of the data block containing the schedule with the switching times.
ENT	IN	INT	I, Q, M, D, L, const.	Number of entries included in the “DBNO” data block.
ERROR	OUT	BOOL	Q, M, D, L	Error status: “1” if an error has occurred
CODE	OUT	INT	Q, M, D, L	Error code **W#16#80A1** : Invalid DB number **W#16#80B1** : DB with number “DBNO” does not exist **W#16#80B2** : DB with number “DBNO” has been generated with the “UNLINKED” keyword **W#16#80D1** : DB too small for the number of entries specified at “ENT” **W#16#80D2** : DB with number DBNO is write-protected **W#16#80D3** : Number of entries “ENT” > 255 **W#16#8100** : Error when reading the system time (SFC1) **W#16#82xx** : Error in entry number xx in the “DBNO” data block

Example

In the program example, the “TOD” (FC1) function is called in “TestFB” (FB1) with all necessary parameters and an error check is performed. If an error occurs, this is indicated via a bit. The input parameters can be changed and the results can be checked in the “VAT_1” variable table created in the project.

NOTICE **For test purposes, the following sample project provides the option of resetting the CPU system time. For this reason, the following steps must only be performed with PLCSIM.**

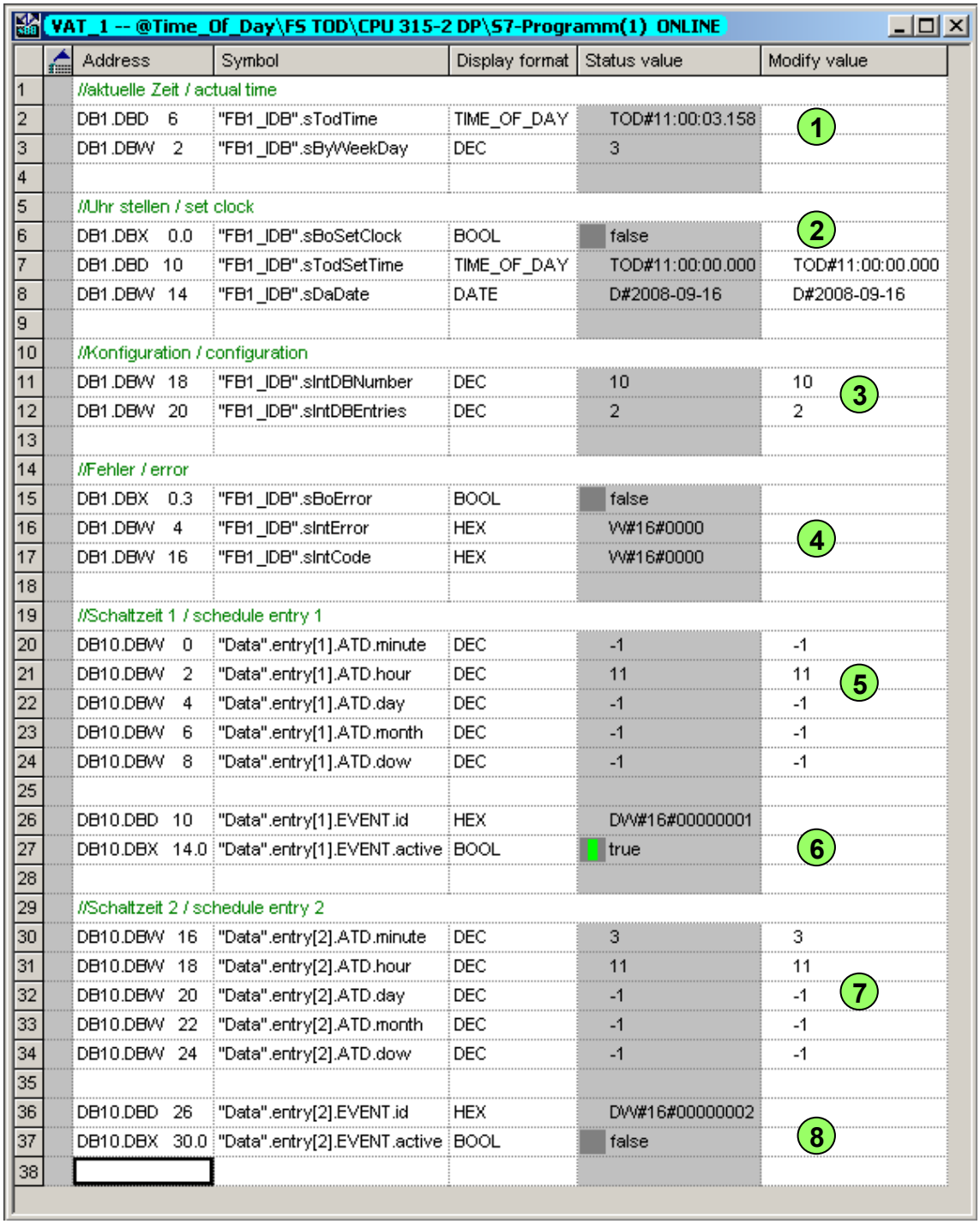
To test the sample project, proceed as follows:

Table 1-10

Step	Action/event
1.	✓ Download the complete station to S7-PLCSIM and open the “VAT_1” variable table in online mode **Result:** • The current system time is displayed (1) • An error is displayed (W#16#80A1) since the current DB number is “0” (4)
2.	✓ Transfer the modify values preset in the variable table to the CPU. (3) (4) **Result:** • The error message disappears since the correct data block is now enabled. • Switching time 1 is configured for exactly 11:00 for each day in each month ➔ ON duration 1 h (5) • Switching time 2 is configured for exactly 11:03 for each day in each month ➔ ON duration 1 min (7)
3.	✓ Set the system clock by setting the “sBoSetClock” binary signal briefly to “1” and then back to “0” (2) **Result:** • The system clock is set to “11:00:00” • The status of switching time 1 at the “Data”.entry[1].EVENT.active output changes to “1” (hour 11 has been reached) (6)
4.	✓ Wait until the system clock displays 11:03. **Result:**

Step	Action/event
	• Now also the status of switching time 2 at the “Data”.entry[2].EVENT.active output changes to “1” (8)
5.	✓ When the relevant times have elapsed, the corresponding switching outputs are reset.
6.	✓ Repeat points 2-5 for any times

Figure 1-6



VAT_1 -- @Time_Of_Day\FS TOD\CPU 315-2 DP\S7-Programm(1) ONLINE

	Address	Symbol	Display format	Status value	Modify value
1	//aktuelle Zeit / actual time				
2	DB1.DBD 6	"FB1_IDB".sTodTime	TIME_OF_DAY	TOD#11:00:03.158	
3	DB1.DBW 2	"FB1_IDB".sByWeekDay	DEC	3	
4					
5	//Uhr stellen / set clock				
6	DB1.DBX 0.0	"FB1_IDB".sBoSetClock	BOOL	false	
7	DB1.DBD 10	"FB1_IDB".sTodSetTime	TIME_OF_DAY	TOD#11:00:00.000	TOD#11:00:00.000
8	DB1.DBW 14	"FB1_IDB".sDaDate	DATE	D#2008-09-16	D#2008-09-16
9					
10	//Konfiguration / configuration				
11	DB1.DBW 18	"FB1_IDB".sIntDBNumber	DEC	10	10
12	DB1.DBW 20	"FB1_IDB".sIntDBEntries	DEC	2	2
13					
14	//Fehler / error				
15	DB1.DBX 0.3	"FB1_IDB".sBoError	BOOL	false	
16	DB1.DBW 4	"FB1_IDB".sIntError	HEX	W#16#0000	
17	DB1.DBW 16	"FB1_IDB".sIntCode	HEX	W#16#0000	
18					
19	//Schaltzeit 1 / schedule entry 1				
20	DB10.DBW 0	"Data".entry[1].ATD.minute	DEC	-1	-1
21	DB10.DBW 2	"Data".entry[1].ATD.hour	DEC	11	11
22	DB10.DBW 4	"Data".entry[1].ATD.day	DEC	-1	-1
23	DB10.DBW 6	"Data".entry[1].ATD.month	DEC	-1	-1
24	DB10.DBW 8	"Data".entry[1].ATD.dow	DEC	-1	-1
25					
26	DB10.DBD 10	"Data".entry[1].EVENT.id	HEX	DW#16#00000001	
27	DB10.DBX 14.0	"Data".entry[1].EVENT.active	BOOL	true	
28					
29	//Schaltzeit 2 / schedule entry 2				
30	DB10.DBW 16	"Data".entry[2].ATD.minute	DEC	3	3
31	DB10.DBW 18	"Data".entry[2].ATD.hour	DEC	11	11
32	DB10.DBW 20	"Data".entry[2].ATD.day	DEC	-1	-1
33	DB10.DBW 22	"Data".entry[2].ATD.month	DEC	-1	-1
34	DB10.DBW 24	"Data".entry[2].ATD.dow	DEC	-1	-1
35					
36	DB10.DBD 26	"Data".entry[2].EVENT.id	HEX	DW#16#00000002	
37	DB10.DBX 30.0	"Data".entry[2].EVENT.active	BOOL	false	
38					

1 2 3 4 5 6 7 8

Technical specifications

Table 1-11

Block	Data
"TOD" (FC1) time-of-day schedule	Required local data : 62 bytes Load memory requirement : 1482 bytes Work memory requirement : 1268 bytes

2 Overview of the Download Files

Download file “31696259_Datum-Uhrzeitfunktionen.zip” contains the ZIP files for the respective functional examples listed below.

Table 2-1

Serial no.	Data block	ZIP file
1.	Determining the calendar day	CalendarDay.zip
2.	24-hour time switch	TSwitch.zip
3.	Time-of-day schedule	TimeOfDay.zip

3 History

Table 3-1

Version	Date	Modification
V1.0	04/24/09	First edition